TASK: PA33
CDRL: AG01
6 February 1996

INFORMAL TECHNICAL REPORT

For

SOFTWARE TECHNOLOGY FOR ADAPTABLE, RELIABLE SYSTEMS (STARS)

Learning and Inquiry Based Reuse Adoption (LIBRA):
A Field Guide to Reuse Adoption through Organizational Learning
Version 1.1

STARS-PA33-AG01/001/02
6 February 1996

Data Type: Informal Technical Data

CONTRACT NO. F19628-93-C-0130

Prepared for:
Electronic Systems Center
Air Force Systems Command, USAF
Hanscom, AFB, MA 01731-2816

Prepared by:
Loral Defense Systems-East
9255 Wellington Road
Manassas, VA 22110

TASK: PA33
CDRL: AG01
6 February 1996

Data Reference: STARS-PA33-AG01/001/02
INFORMAL TECHNICAL REPORT
Learning and Inquiry Based Reuse Adoption (LIBRA):
A Field Guide to Reuse Adoption through Organizational Learning
Version 1.1

Distribution Statement "A"
per DoD Directive 5230.24
Authorized for public release; Distribution is unlimited

This document, developed under the Software Technology for Adaptable, Reliable Systems (STARS) program, is approved for release under Distribution "A" of the Scientific and Technical Information Program Classification Scheme (DoD Directive 5230.24) unless otherwise indicated. Sponsored by the U.S. Advanced Research Projects Agency (ARPA) under contract F19628-93-C-0130, the STARS program is supported by the military services, SEI, and MITRE, with the U.S. Air Force as the executive contracting agent. The information identified herein is subject to change. For further information, contact the authors at the following mailer address:
delivery@stars.reston.unisysgsg.com.

Data Reference: STARS-PA33-AG01/001/02
INFORMAL TECHNICAL REPORT
Learning and Inquiry Based Reuse Adoption (LIBRA):
A Field Guide to Reuse Adoption through Organizational Learning
Version 1.1

Abstract

This document presents a new and innovative learning-oriented approach to reuse assessment and adoption. The approach, entitled Learning and Inquiry Based Reuse Adoption (LIBRA), focuses on highly participative scenario-based techniques derived from the fields of organizational learning and theatrical scripting.

TASK: PA33
CDRL: AG01
6 February 1996

Data Reference: STARS-PA33-AG01/001/02
INFORMAL TECHNICAL REPORT
Learning and Inquiry Based Reuse Adoption (LIBRA):
A Field Guide to Reuse Adoption through Organizational Learning
Version 1.1

Principal Author(s):

Sidney Bailin, Knowledge Evolution, Inc. Date

Mark Simos, Organon Motives, Inc. Date

Larry Levine, Organon Motives, Inc. Date

Dick Creps 2/6/96

Dick Creps, Loral Defense Systems-East Date

Approvals:

Teri F. Payton 2/6/96

Program Manager *Teri F. Payton* Date

(Signatures on File)

REPORT DOCUMENTION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave Blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	06 February 1996	Informal Technical Report

4. TITLE AND SUBTITLE

Learning and Inquiry Based Reuse Adoption (LIBRA): A Field Guide to Reuse Adoption through Organizational Learning, Version 1.1

5. FUNDING NUMBERS

F19628-93-C-0130

6. AUTHOR(S)

Sidney Bailin: Knowledge Evolution, Inc., Mark Simos and Larry Levine: Organon Motives, Inc., Dick Creps: Loral Defense Systems

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Loral Defense Systems-East
9255 Wellington Road
Manassas, VA 22110-4121

8. PERFORMING ORGANIZATION REPORT NUMBER

CDRL NBR
STARS-PA33-AG01/001/02

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Department of the Air Force
ESC/ENS
Hanscom AFB, MA 01731-2816

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

AG01

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Distribution "A"

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

This document presents a new and innovative learning-oriented approach to reuse assessment and adoption. The approach, entitled Learning and Inquiry Based Reuse Adoption (LIBRA), focuses on highly participative scenario-based techniques derived from the fields of organizational learning and theatrical scripting.

14. SUBJECT TERMS

15. NUMBER OF PAGES
107

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	SAR

Data Reference: STARS-PA33-AG01/001/02
INFORMAL TECHNICAL REPORT
Learning and Inquiry Based Reuse Adoption (LIBRA):
A Field Guide to Reuse Adoption through Organizational Learning
Version 1.1

Table of Contents

Data Reference: STARS-PA33-AG01/001/02
INFORMAL TECHNICAL REPORT
Learning and Inquiry Based Reuse Adoption (LIBRA):
A Field Guide to Reuse Adoption through Organizational Learning
Version 1.1

Table of Contents

Data Reference: STARS-PA33-AG01/001/02
INFORMAL TECHNICAL REPORT
Learning and Inquiry Based Reuse Adoption (LIBRA):
A Field Guide to Reuse Adoption through Organizational Learning
Version 1.1

List of Exhibits

Prologue

This document is version 1.1 of *Learning and Inquiry Based Reuse Adoption (LIBRA): A Field Guide to Reuse Adoption through Organizational Learning*. It proposes a new and innovative approach to reuse adoption, focused on techniques emphasizing learning and inquiry. See Section 1, Introduction, for an overview of the approach.

We intend to apply and to continue to develop the approach in the future and are very interested in hearing any feedback you may have about the approach or its presentation in the document. If you would like to impact this work in the future or are interested in applying it, please phone, fax, mail, or e-mail your comments to:

Dick Creps
Loral Defense Systems-East
9255 Wellington Road
Manassas, VA 22110
Phone: (703) 367-1353
Fax: (703) 367-1389
E-mail: creps@stars.reston.unisysgsg.com

Thank you for your interest.

1.0 Introduction

Many approaches to reuse assessment, adoption and planning have been proposed and, to some degree, applied in recent years. These include (among others) the STARS Conceptual Framework for Reuse Processes (CFRP) [CFRP93] and Reuse Strategy Model [RSM93], the Software Productivity Consortium's Reuse Adoption Guidebook [RAG93], the Comprehensive Approach to Reusable Defense Software (CARDS) reuse handbooks (e.g., [CARD92]), the SEI's Reuse Opportunity Analysis Model (ROAM), and the SEI's Capability Maturity Model (CMM).

These approaches have made major contributions to our understanding of the economic, organizational, and cultural aspects of reuse. Particular progress has been made in addressing how to build a business case for reuse, and how to address resistance to the introduction of reuse technology from a technology transfer viewpoint.

As valuable as these perspectives are, however, we view them as necessary but not sufficient. In focusing on visible symptoms, they leave unaddressed the core problem of *individual and institutional beliefs and interaction patterns that can motivate or inhibit reuse*. To motivate people within organizations to adopt reuse-based practices we must do more than present them with new information. Reuse adoption, and the organizational changes that it usually requires, may challenge people to make shifts in deeply-held beliefs (how I think the world is) and values (how I want the world/this organization/my work to be). This document represents an attempt to define a new approach to reuse assessment and adoption that complements existing efforts by directly addressing these beliefs and the interaction patterns to which they are linked.

Imagine a situation where the economic benefits of introducing systematic reuse were conclusively demonstrated to management. Imagine, also, that no new technology needed to be introduced in order to create a systematic reuse program. Now try to imagine all the things that could still go wrong and prevent success of the reuse program.

These factors are the primary focus of this document. It offers tools and techniques to help you identify barriers that might not be openly discussed, or might be hidden to people in the organization. The approach shifts some assumptions about the nature of reuse that underlie many current approaches; this new perspective in turn suggests a wide variety of new tools and techniques for people to use who want to encourage reuse in their organizations.

Our approach, which we have entitled Learning and Inquiry Based Reuse Adoption (LIBRA), is founded on three key principles: 1) a view of reuse adoption as shifts in beliefs and interaction patterns in an organization; 2) an inquiry-based approach to assessment of these dynamics in the organization; and 3) a shift in perspective that views systematic reuse as a form of organizational learning and knowledge creation.

The overview section that follows elaborates on these principles and introduces a set of tools and techniques to support the approach. The remainder of this introduction describes the intended audience, objectives, organization, and recommended usage for this document. The section concludes by briefly describing the relationship of this document to other work.

1.1 LIBRA Overview

The LIBRA approach is founded on, and motivated by, the following principles:

1) ***Reuse Adoption as Shifting Beliefs.*** The importance of beliefs in reuse adoption is by no means a new idea. The transition to a reuse-based approach to software development is often described as a "paradigm shift" or a "change of mind-set." This implies a shift in beliefs. In

the reuse literature there is frequent anecdotal mention of beliefs that create barriers to reuse adoption. (One example is the oft-cited "not-invented-here" syndrome.)

Informally, reuse champions in organizations are well aware of this challenge in reuse adoption. But these aspects of the "practice" of being a reuse change agent—attempts to persuade short-sighted managers, arguments with skeptical engineers or devotees of various methods that claim to solve the reuse problem—are rarely discussed directly in the literature. There have been few attempts to examine the various beliefs that create motivators or barriers to reuse adoption, and even fewer guidelines provided to aid reuse proponents in shifting such beliefs. In particular, little attention has been paid to the quality of the interactions between the reuse proponents themselves and other people in their organization.

2) ***Inquiry-Based Assessment.*** As the LIBRA name implies, we are aiming to define an approach to reuse adoption where ***learning*** and ***inquiry*** become the integrating principles. The essential starting principles of the LIBRA approach are as follows:

— Deeply held patterns of belief are sometimes nearly invisible to those who hold them. The beliefs are often "undiscussable" as well, especially if they touch on issues that are sources of anxiety and uncertainty — the "harsh realities" — in the organizational environment.

— No one can make anyone else shift their beliefs. Managers can attempt to enforce behavior changes, but cannot force changes to the underlying belief structures that motivate behavior. Attempts to do so may actually be self-defeating by creating resistance and strengthening beliefs that are counter to the intended change.

— To encourage desired shifts in underlying beliefs, an approach based on ***advocacy***—persuasion and argument—is often less successful than creating conditions where people can engage in ***dialogue***: reflecting on their own situation and beliefs, openly articulating the reasoning behind their positions, and entertaining different perspectives. These conditions facilitate and encourage a shift of beliefs but still leave people free to make or not make a shift as they choose.

We use the term ***inquiry*** to denote a broad set of skills that help to *elicit* dialogue in interactions. Dialogue is great if all participants are practicing it; it can break down or be disrupted if one or more participants jump back into an advocacy role. Inquiry skills go farther, in that they can serve as ***interventions*** in situations that begin with advocacy, moving them towards the possibility of dialogue.

One way of suggesting the difference between an inquiry vs. advocacy mode of interaction is that in inquiry, participants do not attempt to directly shift the beliefs of other participants, but rather focus on building a shared model of the diversity of their beliefs. In other words, rather than merely "agreeing to disagree" participants endeavor to agree about what they disagree about and *how* they disagree.

Inquiry-based interaction skills are thus an important part of the repertoire for any change agent within an organization. These skills are particularly important for reuse change agents, since reuse is a topic that many people feel they understand from their own experience, and about which they hold strong opinions.

Ironically, many would-be reuse champions, in being vociferous reuse *advocates*, may actually work against their intentions by increasing resistance in their interactions with people who are not convinced about the benefits of reuse. A central aim of this document is to encourage people in this position to experiment with shifting their own beliefs about how to successfully encourage reuse practice in their organizations—in effect, to shift from being

"reuse advocates" to "reuse inquirers."

3) ***Reuse as Learning and Knowledge Creation.*** We believe that one of the forms of reuse "advocacy" that can actually create more resistance in the reuse adoption process is an over-emphasis on "technology push." That is, in applying many useful technology transfer principles to reuse, a gradual tendency has emerged to view organizational adoption of reuse-based practice as *primarily* a technology transfer problem.

   Rather than viewing reuse as something that can be inserted into an organization via traditional technology transfer techniques, we view systematic reuse as an outcome that springs naturally from an organization that has become a ***learning organization***. In a learning organization, learning activities are explicitly supported and managed alongside direct production activities. Systematic reuse is systematic learning about the processes and products of software development, and the codification of that learning in explicit, shared forms that are accessible to the organization.

   The emphasis on making the results of learning explicit and codified is central to systematic reuse. This goes beyond the general idea of organizational learning and involves a more radical shift to the idea of a ***knowledge-creating organization***. A knowledge-creating organization recognizes the creation of new knowledge as an integral part of every work process, and makes the capture of such knowledge central to its mission and business model. Where a learning organization might still treat learning activities as "process improvement" activities, the knowledge-creating organization considers systematic learning as strategic.

A Different Approach to Adoption and Assessment. Because systematic reuse involves a shift towards this vision of the learning-based or knowledge-creating organization, reuse adoption can elicit powerful reactions from people, in the form of both motivation or ***receptivity*** and barriers or ***resistance***. One of the most important tasks of reuse assessment, as a first step in reuse planning for an organization, must be to identify these points of receptivity and resistance, particularly as they are manifested in systemic patterns of behavior and underlying beliefs that support them. The LIBRA approach focuses specifically on discovering these beliefs and interaction patterns by applying learning- and inquiry-based techniques to reuse assessment, reuse adoption, and reuse practice.

This starting point turns upside down many fundamental assumptions that have guided how proponents of reuse try to effect changes in their organization. While diverse in specific content, most current approaches to reuse adoption share a common set of assumptions:

- The initial impetus or movement towards a reuse initiative within an organization results from the efforts of reuse advocates (e.g., technologists) convincing decision-makers (e.g., managers) to invest—an interaction characterized by *persuasion.*
- In order to decide what reuse actions are appropriate, an assessment is performed that involves characterizing the organization via checklists of criteria that can be linked to recommended actions. Whether performed by an external agent or applied internally, this form of assessment is often viewed as evaluative in nature—i.e., as an *audit.*
- Further planning for reuse adoption involves a top-down, structured approach that maps out a plan for overall institutional change towards reuse-based practice—*mandated change.*
- When planning leads to a specific project, the necessary education and transfer of key reuse concepts takes place through *training.*

These four approaches—advocacy by persuasion, assessment by audit, change by mandate, education by training—occur at different parts of the adoption process but are closely interwoven in

nature. They reflect values and approaches to organization change and technology transfer much broader in scope than the reuse field itself. As change agents in these other fields have discovered, a key problem with such approaches to advocacy, assessment, planning, and technology transfer is that they provide no techniques for surfacing and addressing the many hidden barriers to change. In fact, they can actually increase certain dynamics of resistance while not revealing and exploiting potential motivating forces for change. In some important ways, then, these methods may inhibit shifts towards a learning and knowledge-creating orientation.

Assessment Through New Interactions. Unlike traditional assessment activities, the LIBRA approach involves creating situations where new kinds of interactions can occur that can increase an organization's receptivity to learning and knowledge creation. This leads naturally to an incremental, decentralized form of reuse self-assessment for people in software organizations. (Self-assessment, in this context, means assessment activities carried out by people within the organization, without outside intervention.)

An example of a new interaction might be a group of engineers working on similar subsystems on multiple projects, who are given a chance to get together and compare notes on their respective approaches. The people brought together in these situations might only rarely have the chance to interact as part of the ordinary workflow of the development environment. This is because, from a production standpoint, there are no direct work-related hand-offs between these people; in fact they may be not only on different projects but in different parts of the organization altogether. The reason for bringing them together is to support organizational learning.

New interactions are themselves a form of organizational self-assessment; that is, they can generate knowledge about the organization's potential motivators, receptivity, and barriers to reuse. The interactions may directly create new opportunities for reuse and may help to introduce new concepts and practices to participants.

The types of assessment activities consistent with this approach are:

- *Informal*: do not require a big learning curve;
- *Small-scale*: can be initiated with a single meeting of a few people;
- *Incremental*: can be grown in small steps, with clear value delivered at each step;
- *"Populist"*: at least in principle, can be initiated by motivated people at any hierarchical level within an organization, not just by upper-level decision-makers;
- *Self-assessment based*: are owned and tailored by and for the specific individuals and organizations involved.

This self-assessment process has several major benefits:

- It helps identify a comprehensive set of motivators and barriers to reuse practice within the organization.
- It avoids introducing new barriers that can result from "technology push" (and more generally, an approach based on advocacy).
- It can help to identify new reuse activities that could be initiated by the participants in the assessment. Many of these possible next steps are small, local initiatives that would be missed in large-scale, top-down reuse planning.
- It gets people immediately practicing skills that are essential for reuse activities such as

domain engineering.

This approach allows conversations about reuse to flow seamlessly into assessment, planning, and education. Assessment is not done by external audit against pre-established readiness criteria which then leads to a menu of action options, but instead becomes a highly participative, incremental, self-governing process. It includes education through direct practice of interaction skills essential for sustained reuse, and it already begins to increase an organization's readiness for reuse. The ultimate goal is to identify behaviors you can employ to initiate and sustain successful reuse projects.

Integrated Conceptual Tools for Implementing the Approach. To support the principles and features of the LIBRA approach, we have selected, adapted, and integrated several conceptual tools. These tools stem from a common, learning-oriented view of both reuse adoption and reuse practice. Each tool helps to improve skills for inquiry, reflection, and dialogue. They are integrated into a tailorable, structured self-assessment framework that can be extended to include other techniques as well. There are four major tools, described in detail in Section 3, as well as several less prominent tools that are discussed briefly in Section 6. The major tools are:

- ***Dramatic Scenarios***: Our approach uses scenarios, as opposed to formal process descriptions or other techniques, as a basis for describing individual and organizational interactions. The types of interactions that characterize reuse adoption are best thought of in terms of scenarios because they involve people, not only in light of their structural roles within the organization, but also their beliefs, values, and personal interests as stakeholders. Reuse change agents need to be able to "read" these situations clearly. Scenarios help develop the required observation and reflection skills.

- ***System Diagrams***: We use representations called *system diagrams* as one way of informally illustrating interaction patterns observed within scenarios. They are pictures of how a series of interactions (and perhaps forces that prevent interactions) link together into a "system" that has persistence (it continues over time) and resistance (it cannot easily be changed). Resistance can be "stubbornness" for patterns we want to change; "health" or "robustness" for patterns we want to remain in place. We provide examples of interaction patterns at both the micro level (e.g., dramatized or idealized conversations) and at the macro level (larger strategic pictures of business setting, market forces, organizational structure).

- ***Belief Maps***: We use *belief maps* to describe the relationships between diverse beliefs that individuals hold on a common topic. An individual belief is often an underlying theory that a person uses in evaluating proposed plans or guiding their own actions. Beliefs exist within a network of supporting and conflicting beliefs. "Belief mapping" involves seeing beliefs in relation to each other. It is a significant reflective step when you are able to see your own beliefs within the map and understand their relationships with other beliefs.

- ***Inquiry Techniques***: We present additional concepts and techniques that help to foster new interactions. One of these is the *Ladder of Inquiry*, a framework for assessing the level of inquiry within interactions. A common element of these techniques is that they provide ways for keeping conversations focused on areas of interest while simultaneously "cooling down" the interactions so that people can reflect and listen better, and allow "undiscussables" to be discussed. The techniques for inquiry are essential for deriving the full value of the scenarios, system diagrams and belief maps.

Together, scenarios, system diagrams, belief maps, and inquiry techniques can be used as an integrated, flexible toolkit for identifying important beliefs and interaction patterns and understanding how they may inhibit or enable reuse in your organization.

1.2 Document Audience

Who we think you are

This document is addressed specifically to ***reuse proponents***. We assume you, the reader, have had some exposure to the language and concepts used in the software reuse researchers' and practitioners' community. Based on this exposure, you believe that a systematic approach to software reuse is a sound approach to improving the way software development is done today. You are interested in exploring ways of helping an organization in which you are involved to make the transition towards a systematic reuse-based approach to software development.

> Caveat: This document is not intended to directly persuade readers about the benefits of systematic reuse. If you consider yourself a reuse skeptic as opposed to an proponent, be aware that the document is not targeted directly towards you. We have cogent arguments to make to you on behalf of reuse but we will not make them here! On the contrary, this document is unashamedly "preaching to the converted," in that it is intended as a kind of "field guide" to assist reuse proponents.[1]

Your organization. We intend for this document to be useful within government (including DoD) and contractor organizations. Toward this end, we have tried to include examples relevant to these settings. The extended scenario in Section 4 addresses a "typical" contractor environment. However, we believe the LIBRA approach is applicable in a broad range of settings. The document makes few assumptions about the type of organization in which you work, other than that it is an organization that acquires, develops, and/or maintains software-intensive systems as a critical part of its business operations. This could include:

- A DoD software acquisition organization;
- A contractor organization that develops, integrates and/or maintains systems under government contracts;
- A commercial software product company that sells shrink-wrapped software to mass-market end-users;
- A consulting firm providing engineering services or strategic consulting to large software organizations; or
- A large organization (e.g., a bank, a manufacturing company) that develops and uses numerous custom applications to support its in-house operations.

Your position. We have designed this document to be usable, in principle, by a person at any level within an organizational structure — "reuse guerrilla tactics," if you will. Thus the document makes few assumptions about your role or position within the organization, except that you probably have a job description more precise than "Reuse Proponent at Large." For example, you could be:

- An acquisitions manager for a DoD organization with responsibility for multiple large systems;

---

1. To be precise, rather than preaching to the converted, we are suggesting to the converted that there are options other than preaching (trying not to preach too much ourselves in the process).

- A technology advisor for a line-of-business manager;
- A project manager who would like to try reuse;
- A senior line-of-business or division manager concerned about maintaining competitive advantage;
- A programmer interested in the technical problems of reuse, and with ideas of your own about how to improve reuse within your organization; or even
- A consultant in the position of transferring reuse concepts to a client organization.

Not all these roles fit with the usual understanding of the term "reuse advocate," which is one reason we have chosen the term "reuse proponent" instead. If you support the goals of reuse yet sense that you have little access to decision-makers within the organization, you are unlikely to see yourself as someone who can make a difference. We would like this document to encourage people in such situations to identify specific actions they can take, however small, to move themselves and the organization forward. Conversely, people in decision-making roles need to be aware that simply mandating reuse within the part of the organization under their authority is not an effective strategy.

> Exercise: Who do *you* think you are? Take a moment to describe yourself in terms of our audience assumptions. How would you characterize your beliefs and values about reuse (as you understand it)? Are you a reuse proponent, a skeptic, neither, both? How would you characterize your organization and your role or position within the organization?
>
> Do you have a sense that your personal initiative could effect changes towards more systematic exploration of reuse opportunities within the organization?
>
> Can you recall an interaction you had with someone in the organization who did not know the reuse "lingo"? With someone whose beliefs appeared to present a barrier to reuse within the organization? Were these interactions successful in your opinion? Why or why not?

Who we think we are

The authors bring varied backgrounds to this document. First and foremost we have drawn on our own experiences in initiating reuse efforts in varied contexts: as developers in contractor and commercial software organizations; in contract research and development; and as external consultants. We also have had the benefit of a co-author with considerable experience in the fields of organizational development, organizational learning, and change management. Through exposure to these diverse viewpoints we have undergone considerable changes in our own thinking about the nature of reuse, reuse planning, and reuse adoption, and have come to see "reuse adoption" as the challenge of transforming software organizations into learning organizations.

1.3 Document Objectives

With the concepts and techniques presented in this document you should be able to see familiar situations with "new observer" eyes. After reading and using the document, you should be able to:

1) Describe a vision of systematic software reuse appropriate to your organization's needs, goals, and capabilities;
2) Identify key roles and the interactions among them that are necessary for reuse;

3) Recognize individual and organizational belief patterns that serve as obstacles or enablers to reuse;

4) Recognize your own beliefs within that repertoire of patterns and reflect on how those beliefs affect you in your role as a reuse proponent;

5) Identify the motivators and barriers to reuse adoption within your organization — e.g., barriers that may be missed by methods focusing exclusively on economic issues such as return on investment, or on ideas of resistance based on traditional technology transfer notions;

6) Identify possible next steps for improving your organization's reuse capability;

7) Catalyze conversations to promote systematic reuse within your organization;

8) Apply organizational learning principles to the way reuse is explained, motivated, and practiced in your organization;

9) Recognize, anticipate, and avoid known pitfalls in reuse adoption, e.g., patterns for resisting, sabotaging, or undoing change efforts;

10) Use the document to reflect on your progress and that of your organization on a recurring basis.

1.4 Document Organization

The document is organized into the following sections:

- Section 1 (this section) provides an overview of the LIBRA approach, describes the intended audience and objectives of the document, offers a brief overview of how to use the document, and describes the relationship between this document and other work.
- Section 2 provides historical background for our approach. It explains where the reuse adoption field is today, how it got there, and why the time is ripe for approaches such as LIBRA.
- Section 3 explains core LIBRA concepts and describes the tools and techniques used extensively throughout the remainder of the document.
- Section 4 presents a detailed fictional case study in scenario form, along with a set of commentaries analyzing the scenario. The case study and commentary offer insight into a variety of adoption barriers, how they can be identified and analyzed, and to some degree, how they can be overcome. It also provides examples of how LIBRA tools can be used in practice.
- Section 5 explains how it is useful to view learning-oriented reuse practices in terms of a network of interactions. It then offers some perspectives on interaction patterns that are characteristic of "reuseful" organizations, both in terms of reuse processes and the interactions among reuse-oriented roles.
- Section 6 presents additional concepts and techniques that can be used to discover and analyze interaction patterns and plan a transition to more reuseful patterns. It also provides further guidance for how to use some of the core tools (e.g., how to develop and use dramatic scenarios).
- A References and Recommended Reading section includes all references cited in the document, as well as other resources we recommend if you want to learn more about key topics.

Readers who feel they have sufficient background in reuse or technology adoption may wish to skip Section 2 and proceed directly to Section 3 to learn about LIBRA tools. You could even skip to Section 4 and begin reading the scenario while referring back to Section 3 as necessary to gain insight into the tools. We believe there is a sound logical flow from the beginning to the end of the document and recommend reading it in that order, but encourage you to engage in whatever learning style is most comfortable for you.

1.5 How to Use This Document

We have tried to design the LIBRA approach so that it can be applied by anyone within a software development organization who wants to increase receptivity to a reuse-based approach. Our hope is to put something very practical in people's hands. Some aspects of this document could almost be read as "crib sheets" for the reuse proponent in the trenches—hence our terming this a "field guide."

This document will be most effective if it is read, then actively *used.* It is intended to help you build and evolve an organizational knowledge base to support and sustain reuse. Readers are encouraged to treat the various tools as starting points for constructing their own descriptions of current and desired behaviors and identifying meaningful next steps they can take.

One process (among many) for how this document can be used is as follows:

- One or more would-be reuse proponents read the document, and do some small-scale (perhaps individual) inquiry exercises to see how the process works.
- If this small group decides the approach can be of value to their organization, they design and perform a customized LIBRA assessment within some broader group. They read the scenario case study presented in Section 4, and apply the theatrical scripting techniques outlined in Section 6 to interpret the scenario, modify or adapt it, or (perhaps with outside facilitation) develop a new one, in order to describe reuse-relevant patterns of behavior and beliefs in their organization. They define methods for gathering data to refine and enrich the description.
- They obtain approval from management to expand the scope of the group beyond the original members. They analyze this selected scope, yielding a more refined description.
- They compare the description to a tailored set of archetypal patterns (both those inhibiting and supporting reuse) to identify potential problems and desired improvements.
- They identify appropriate intervention steps and work with other members of the organization to apply them.
- The organization continues to iterate the process, and to re-invoke it whenever systematic learning seems to be needed. A natural by-product of this learning is an evolving knowledge base of organizational information, including interaction patterns and interventions that were designed to work for the organization (or are known not to work, as the case may be).

Besides the specific techniques we have introduced, we have generated example material (most prominently, the scenario in Section 4) that reflects our own personal opinions, beliefs and biases. This is as it should be: the techniques are designed to elicit precisely this kind of data. Readers should be aware, however, that the specific example scenarios and interpretations we have provided should not be treated as an integral and inseparable part of the assessment techniques themselves. The examples are intended only as seeds or catalysts to help you create and capture knowledge that is most relevant to your own organization.

Terminology. We outline underlying concepts and theory where this seems essential to enable effective use of the techniques. Elsewhere in the document we have employed unusual structure and format to best convey the approach (for example, use of conversations or dialogues). Some of the techniques described and terminology used may be unfamiliar to software engineers, though we attempt to define terms borrowed from different communities. We ask readers' tolerance for any periodic lapses into "dialect from a different village," and encourage them to use such occasions to practice their own inquiry skills.

1.6 Relationship to Other Work

The techniques described in this document are complementary to and can be used in combination with other approaches to reuse assessment, adoption, and practice. In particular, we believe that the LIBRA approach can add substantial value when used in conjunction with existing approaches such as the Software Productivity Consortium's reuse adoption process, the CARDS reuse partnership approach, or even the Software Engineering Institute's Capability Maturity Model. However, LIBRA does not rely on any other methods or approaches and, we believe, can be applied by itself with good results.

Relationship to STARS. This document was developed by the Loral Defense Systems-East STARS team and reflects many of the reuse perspectives of the STARS program, including a strong belief in and commitment to systematic, product-line approaches to reuse. In addition, the authors bring their unique perspectives to this work, including experience in organizational change management and theatrical scripting as well as software engineering and reuse. These perspectives from multiple disciplines have been integrated to produce what we believe is a new and innovative approach that nevertheless retains its STARS roots.

ReuseWorks Technologies. ReuseWorks is a comprehensive suite of reuse concepts, processes, methods, and tools developed under the auspices of the ARPA/STARS Program and the Loral Defense Systems-East team. Many of the people involved in the ReuseWorks effort are also involved in developing LIBRA. Thus, the LIBRA approach holds a strong affinity for many of the ReuseWorks technologies. In particular, we have drawn heavily from the STARS Conceptual Framework for Reuse Processes (CFRP), a high-level process model that identifies a comprehensive set of processes and their interactions suitable for describing reuse within software organizations. Section 5 of this document describes how the CFRP can be used to help identify and change networks of reuse-relevant interactions within an organization.

Another ReuseWorks technology that is clearly relevant to LIBRA is the Organization Domain Modeling (ODM) domain engineering method. In particular, the ODM domain planning and scoping processes provide useful techniques for negotiating the technical scope of reuse-related interactions. ODM's potential contribution to LIBRA is described briefly in Section 6.

Although this document complements and adds value to the ReuseWorks suite of products, it can be used independently of them in a stand-alone way or to augment other reuse assessment or adoption approaches. We believe the LIBRA approach will prove useful to a broad range of researchers, proponents, and practitioners in the software engineering and reuse community.

2.0 Background

The LIBRA approach to reuse adoption is a product of insights gained from the current state of practice in several fields. Though it addresses different aspects, it depends on the progress made in these other areas, and in that sense builds on and complements this other work. It is unlikely that LIBRA could have been formulated earlier in the history of the reuse field.

In this section, we provide a thumbnail history of approaches to reuse adoption. We then provide motivation for a view that reframes reuse adoption in terms of knowledge creation and organizational learning.

2.1 Approaches to Reuse Adoption

For many years, researchers and advocates of reuse felt like voices crying in the wilderness. In the defense systems world, the acquisition process and contractors' environments erected significant barriers against reuse. In commercial software organizations, pressures of short-term planning and investment cycles, programmers' culture, and technical obstacles made systematic reuse a hard sell. In this generally hostile climate, reuse change agents faced substantial challenges in trying to influence technology and business shifts in their organizations.

An informal survey of approaches to reuse adoption over the last 10-15 years reveals a progression of ideas in people's thinking about how organizations move towards reuse:

- ***Phase 1: Naive enthusiasm.*** In this phase technical merits of a reuse-based approach are dominant in the mind of the reuse proponent. Fascination with reuse can arise naturally out of a software developer's daily practice. It is also natural for this first enthusiasm to take the form of a primarily technical vision--if only the language provided the right support, the proper components were in place, etc. Reuse is just the "right way" to do things; and anyone who is not thick-headed or entirely self-interested will see its value if the technical case is made clearly enough. This idealism can often lead to a simplistic approach to the technical problems involved. However, people willing to take such a straightforward approach are probably responsible for much of the reusable software that has actually been written.

- ***Phase 2: Recognizing non-technical barriers.*** In this phase the proponent, chastened from a few encounters with resistance to reuse, rethinks his/her position. The first "loss of innocence" comes from the experience of trying to introduce reuse within the organization at a non-trivial scale (i.e., at a scale visible to management). Here non-technical issues begin to surface: not-invented-here syndrome, short-sighted management practices, a "get the job out of the door as quickly as possible" mentality, etc.

  The response within the reuse researchers' and practitioners' community is increased attention to non-technical issues. The attention, however, is still grudging.The attitude is that we must deal with these annoying non-technical issues in order to motivate what is still viewed primarily as a technical change, adoption of new technology.

- ***Phase 3: Need for a business case.*** After enough encounters with non-technical issues, it becomes clear that institutionalizing reuse is a strategic business decision that must be motivated, like other decisions of that kind, with a business case. If a clear cost-benefit case can be made for reuse and the perceived risk is sufficiently low, management will decide to initiate it. (This presumes that the decision to initiate a reuse program can be motivated entirely on economic grounds.) In the DoD community, recognition of the need for software acquisition reform would seem to belong to this phase; in the commercial arena, there is a greater awareness of the need for clear reuse-supportive funding models.

It has taken the reuse field many years to get to the point where we understand how such business cases must be made. This is a major step forward. In many cases, the ability to create a compelling business case is a necessary step to get *commitment* and *action* towards a reuse initiative. However, there is nothing that guarantees that such initiatives will actually succeed; often they do not. (The business case attempts to prove that *if* the initiative succeeds there will be significant return on investment).

- ***Phase 4: Reuse as technology transfer.*** Acknowledging that even a compelling business case for a reuse initiative may be rejected, or that an initiative may fail despite sound technical and business planning, a fourth phase in approaches to reuse adoption has emerged through recognition of the many technology transfer issues involved. By framing the problem in this way, it is possible to bring to bear a vast base of theory and practical knowledge about technology transfer and innovation diffusion. There has been a great deal of excellent work done in recent years "transferring the knowledge" of the technology transfer community to the reuse community, particularly by the SEI. More recently, the work of Geoffrey Moore on high technology market forces has been increasingly influential [Moor91].

The "Reuse Wave." Attention to both the business case and technology transfer aspects could be said to represent the current state of the art in work on reuse adoption. As a result of these and other factors, a "reuse wave" is now building within government and industry. Awareness of reuse-based/architecture-driven/product-line concepts is steadily increasing. Many organizations are "hyping" the importance of these ideas. High-level efforts are underway within the DoD to transition to a domain-specific reuse-based, architecture-driven product-line approach to software development. Technologies such as client-server architectures, open systems, Web-based information services, ORB-based components, OO analysis and design methods, and increasingly interoperable platforms and applications appear to be removing significant technical obstacles to reuse.

These recent developments create both exciting opportunities and new kinds of dangers. In the "bad old days," failure to engender interest in reuse was a missed opportunity, not usually a disaster (for the reuse field, that is; on occasion it was a disaster for the organization concerned). With the increasing expectations being placed on reuse, more and larger initiatives are being funded. Unfortunately, these trends (even apparently positive developments such as the growing momentum behind the idea of reuse) may substantially increase the risk of using the current repertoire of "advocacy-based" adoption strategies. Here are some reasons:

- Issuing a "mandate" when you do not really have authority is ineffective. Issuing a mandate when you do have authority may set forces in motion that achieve near-term results but yield significant failures farther downstream because the mindset of the workers has not changed.

- Accurate but incomplete assessments may lead to planned actions which will encounter unforeseen forces of resistance and inertia. As reuse gains more recognition and support, it becomes harder to blame failed efforts on lack of resources or general awareness. Other barriers to adoption become more apparent. If underlying reasons for breakdowns are not addressed, failure of the initiative is much more likely.

- Barriers may arise not at the beginning but well into the initiative, when considerable resources have already been expended, expectations have become inflated, and the personal credibility of some key players depends on sweeping issues under the carpet. The consequences of such failures can be immense, because the resources once committed will generally not be invested again; the field is soured for future efforts.

- As reuse planning expands in scope, the consequences of failure can increase dramatically. Five-year, enterprise-wide reuse plans require considerable buy-in and commitment. Condi-

tions for such buy-in are rarely in place and declared commitment often masks unstated resistance or represents good intentions from managers who lack the consistency to stick with a plan over the long term.

- Reuse initiatives may become magnets for other agendas (including specific technologies or other improvement initiatives). For example, the reuse program may present an opportunity for one group to impose a standard, generic architecture on other groups. If the architecture has not been engineered for widespread reuse, it will often contain rigid or unreliable elements drawn from particular assumed contexts of use. Results in some cases can be worse than if no reuse effort had been initiated at all.

A New Approach. To circumvent these technology transfer traps, alternative strategies are needed to augment advocacy-based adoption approaches. The LIBRA approach is an attempt to fill this need. It is founded on the idea that sustained adoption of new concepts and technologies occurs only in a climate that is receptive to change. Such a climate is formed not simply by thrusting new ideas at people, but by identifying and resolving existing positions, beliefs, and interaction patterns through learning and inquiry.

We believe the time is ripe for adoption approaches such as LIBRA. Trends in the business environment are heightening receptiveness to learning-based approaches. By understanding the link between systematic reuse and these broader business trends, reuse proponents will be better able to align themselves with forces for change in their organizations. At the organizational level, we see steady movement towards business process reengineering, restructuring of organizations around core competencies, and the rise of virtual enterprises. The software industry of the future may be built around small-scale, component-like companies, providing focused sources of expertise within adaptable networks of contracts and partnering relationships. These trends pave the way for business models that will both encourage and benefit from systematic reuse of software and knowledge assets.

2.2 Reuse As Technology Transfer "In Reverse"

Framing reuse adoption as a technology transfer problem makes sense to the extent that reuse-based practice involves acquiring or developing a specific technology or set of technologies (e.g., object-oriented development methods, a software repository infrastructure, application generation techniques).

And yet, systematic reuse is often spoken of informally as a kind of shift of mind-set, a paradigm shift in beliefs and values, a change in the way software is developed. When we apply models from technology transfer, we implicitly adopt a metaphor which says that reuse is something external (a technology) that must be "transferred," or "inserted into," or "adopted by" an organization. In doing so, we frame the task in a way that often invites resistance and creates unnecessary obstacles. It diverts the focus of reuse initiatives away from the software assets and knowledge developed within the organization itself; i.e., the technology which that organization could be "transferring" more effectively, both internally and outside the organization.

The LIBRA approach is founded on shifting this metaphor. In this alternative perspective, reuse is viewed as capturing and more effectively disseminating the technologies (or more generally, the competencies and knowledge) that emerge *within* an organization; the technology to be transferred is the expertise the company already has. Where conventional technology transfer uses the image of external technology moving *into* the organization, the LIBRA approach encourages technology that is buried and hidden *within* an organization to be made more widely visible throughout, as well as outside, that organization.The most significant transfer in systematic reuse is thus from the "inside out," i.e., from the organization to the surrounding environment, rather

than from the "outside in." To emphasize this shift of perspective we say that reuse is "technology transfer in reverse."

The position can be stated in the following way:

> This company (division/group) knows a lot about building {X} systems. Up until now management has counted on informal ways of transferring this knowledge from developer to developer. If we provide supporting infrastructure to make this knowledge transfer a legitimate and valued part of everyone's job, we will be better able to capitalize on this corporate expertise.
>
> We may even be able to market this expertise externally in ways that are not yet evident to us. Right now we think of ourselves as primarily a product (service/contract)-driven business. If we make our *knowledge* a core part of our business identity, we will be better able to grow new services out of existing products and new products out of existing services, and to use our experience as leverage for new contracts.

This approach places reuse proponents more in the position of facilitators than persuaders, allies for technical people "in the trenches" who may have had difficulty communicating their insights to their own managers. This can help reverse the power relationships that are typically perceived when proponents try to get reuse initiatives started.

Reuse, often presented to software developers in a way that leads them to believe it is just another management fad imposed from above, subject to abrupt and arbitrary changes, can instead be presented as (and can be) a way for developers' knowledge to be legitimized and systematically supported. The approach thus invites commitment, buy-in, and ownership on the part of those whose involvement is crucial to the success of reuse efforts.

Adoption of technology that supports or facilitates reuse is instrumental, but not central, to this change process. Reuse support technology makes it easier for people to share knowledge more effectively with each other. But the primary change is organizational, involving the transfer of knowledge and the establishment of new channels of communication within and across groups in the organization.

Reuse as technology transfer in reverse is a central foundation of the LIBRA approach. It relies on a different underlying metaphor — reuse-based software development as a form of *knowledge creation.*

2.3 The Knowledge-Creating Organization

It comes as no surprise that organizations can be thought of as producing and consuming knowledge. A focus on knowledge production is not new for R&D groups or for university settings. But in most business settings, the generation of new knowledge within the work process has traditionally been viewed as a by-product of the core process, which is providing a service, fulfilling a contract, or developing and marketing a product. The idea that knowledge creation is central to the daily activities of all or most departments and individuals in the workplace is quite new. Escalating business pressures are forcing organizations to rely more on producing new knowledge, applying it systematically, and expanding their capacity through learning.

Ideally, the knowledge captured would be unique, strategic, or special to the organization. In many cases there are opportunities to propagate this knowledge beyond the boundaries of the organization (i.e., the "technology transfer in reverse" principle outlined above). Systematic learning and reuse of knowledge empowers an organization to transfer technology competencies to the marketplace as new sources of competitive advantage. It enables strategic shifts from a product-, contract-, or service-based business model to model of a "knowledge-creating" organization [Nona91, Nona95], in which the generation and codification of new knowledge makes it possible to adapt to new challenges and opportunities more rapidly and flexibly than before.

This trend is of particular relevance to software development (and, more generally, software-intensive) organizations. Software itself defies the traditional categories of service versus product. Software development, maintenance, and operation remain significantly oral, informal, and non-hierarchical in nature. This creates dilemmas for codifying, capturing, and communicating software knowledge from project to project, person to person, department to department, and company to company.

How is Knowledge Created?

Three concepts are useful to recognize how common interactions create knowledge:

- ***Knowledge is more than information.*** Most interactions are not just exchanges of information; they actually involve creation of new knowledge. However, this often happens so automatically or at such a fine-grained level that we do not recognize the interactions as knowledge-creating activities. For example, in the software field we often overlook or discount the kind of knowledge created when people in a software group recognize that they are designing similar interfaces for different products, or that a customer proposal has close similarity to one bid two years earlier.

- ***Knowledge is socially situated.*** New knowledge is created when people interact. That is, knowledge creation is not merely a matter of identifying and debriefing isolated experts within the organization. Knowledge frequently resides within sustained networks of interactions. This is a lesson learned in many software company acquisitions, where a software system treated as a tangible (and transferable) asset proves to be highly dependent on the web of informal knowledge held by the development or maintenance team.

- ***Knowledge is often invisible.*** The most important things we know are often those things we know so well that we don't know we know them (i.e., "second-nature" knowledge). Knowledge creation accelerates when this invisible knowledge is raised to awareness. This problem is central to systematic reuse: the attempt to reuse an artifact created for one system context usually reveals many hidden contextual dependencies that were not noted by the original developers.

The Knowledge Evolution Grid shown in Exhibit 1 provides a useful framework for understanding how knowledge can be systematically captured and made more accessible in an organization. It shows the migration of knowledge along two different dimensions, from private to public and tacit to explicit forms. Public and explicit knowledge is essential to (and a result of) systematic learning and reuse. Section 6 describes how the grid can be used as a tool for reuse assessment and improvement.

2.4 Organizational Learning

If reuse change agents are really in the business of transforming software organizations into knowledge-creating organizations, strategies evolved to support traditional technology transfer

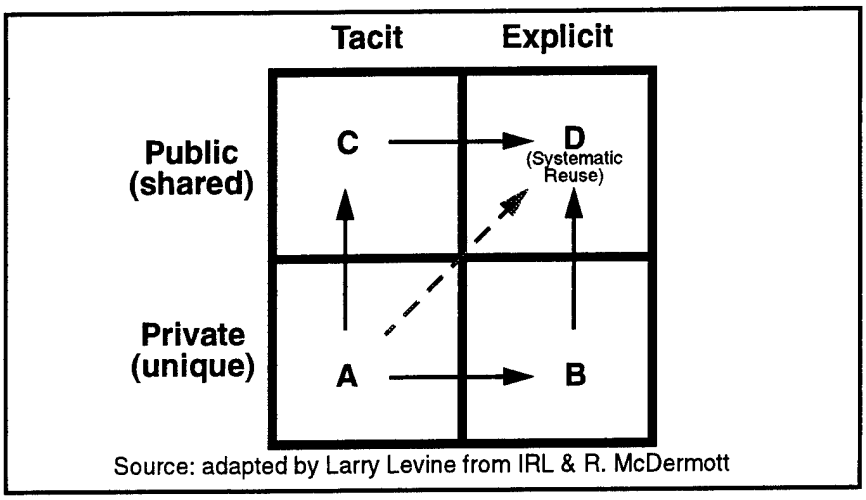


Exhibit 1. Knowledge Evolution Grid

will only be of limited use. Techniques are needed that help make motivators and barriers to adoption more visible. Some of these motivators and barriers correspond to the factors addressed by a good business case analysis or the classic resistance recognized in technology transfer and diffusion. Other motivators and barriers encountered by reuse change agents closely mirror the experiences of people involved in the fields of organizational development, change management, and organizational learning. In particular, a reuse change agent must address the organization's receptiveness to systematic learning and knowledge-creation.

There is a large body of relevant work that can be leveraged from these fields. Reuse can be viewed as a specialization of organizational learning. Just as previous work tailored general technology transfer guidelines to apply to reuse, general organizational learning principles can be tailored to the special conditions of software organizations.

The concepts and techniques of organizational learning are central to the LIBRA approach. We have selected and integrated concepts and techniques from several sources that we believe will be most useful and most readily learned starting with little or no background in organizational learning principles or organization change management. We have also interpreted and grounded these general tools to suit the needs of software development organizations concerned with reuse.

The Fifth Discipline work of Peter Senge and Innovation Associates [Seng90, Seng94] is the most accessible and widely known body of work in the organizational learning field. Senge's ideas concern "the art and practice of the learning organization" via five inter-related disciplines: shared vision, personal mastery, systems thinking, mental models, and team learning.

Although the disciplines of ***shared vision*** and ***personal mastery*** are not focal points of this document, the LIBRA approach offers a good basis for building a shared vision of reuse within an organization. LIBRA is consistent with the philosophy of building vision through interactions and communications at all levels, rather than having it set by top-level management and propagated down through the hierarchy. Similarly, the LIBRA approach challenges reuse proponents to strive for personal mastery through the development of inquiry skills.

LIBRA focuses more directly on the other three disciplines. Our use of scenarios and system diagrams is intended to foster Senge's discipline of ***systems thinking***, while belief mapping draws from the disciplines of ***mental models*** and ***team learning***. The tools for inquiry-based conversa-

tions owe a more direct debt to one of Senge's major influences, Chris Argyris. (For example, our Ladder of Inquiry is an adaptation of his concept of the Ladder of Inference [Argy91].)

A thorough introduction to these ideas is beyond the scope of this document, but later sections provide sufficient additional background to guide use of the techniques presented.

2.5 Learning-Oriented Assessment

It is no trivial matter to transition an organization to a knowledge-creating orientation. It is a profound change that affects people at all levels of the organization, and impacts everything from organization structure and market position to how the organization defines its core mission. Approaching organizations as knowledge-creating and knowledge-sharing enterprises requires innovations in organization design and change management.

On the other hand, large-scale institutional change, reorganization, or reengineering strategies have not been the most successful ways of fostering a learning orientation within organizations. One advantage of our approach is that it helps people to identify small-scale initiatives that can get new kinds of learning interactions going. Usually it is the people "in the trenches" who are best able to recognize such opportunities. Frequently, new interactions that foster systematic learning in an organization can begin with very small, local, low-risk initiatives.

Our approach goes farther than recommending small-scale adoption efforts; we also see the assessment process itself as something that can be decentralized, localized and performed incrementally. For some organizations, even the cost of a thorough reuse assessment involving surveys, strategic planning, and interviewing may be a substantial barrier. This does not imply that all change can be effected incrementally, with no broad, strategic commitment from management. Certain improvements will only be achieved with up-front investment of resources well above the "noise level" for the organization.

Because systematic reuse can involve major shifts in beliefs and interaction patterns, reuse adoption can elicit powerful reactions from people, in the form of both motivation or receptivity and barriers or resistance. One of the most important tasks of reuse assessment, as part of either a small-scale or large-scale adoption effort, is to identify these points of receptivity and resistance, particularly as manifested in systemic patterns of behavior and underlying beliefs that support them. The LIBRA approach focuses specifically on discovering — and facilitating shifts in — these beliefs and interaction patterns by applying learning- and inquiry-based techniques to reuse assessment, reuse adoption, and reuse practice.

3.0 Core LIBRA Concepts and Techniques

This section presents a core set of concepts and techniques that are useful in assessing an organization's receptiveness and resistance to systematic learning and knowledge-creation.

Core Principles

The core LIBRA techniques:

- Foster interactions that balance inquiry and advocacy;
- Help to identify incremental assessment and adoption steps, rather than large-scale planning efforts requiring significant up front investment;
- Emphasize qualitative self-assessment that is motivated, performed, and tailored by members of the organization rather than by an outside "auditing" organization;
- Emphasize identification of the points of reuse receptiveness and resistance that are particular to each organization;
- Help to reinforce the view of reuse as systematic learning and knowledge creation.

Who, What and How

Three critical elements can be shifted to create conditions for new kinds of interactions that increase an organization's readiness for systematic reuse:

- ***Who* is in the room?** By creating situations where people who don't normally talk with each other can exchange ideas and experiences, some habitual patterns can be left behind and new ones discovered;
- ***What* are people talking about?** By keeping the focus of the conversations on the specifics of software systems and development processes, but with the added "twist" of looking for potential reuse opportunities, new patterns of interaction can be created without making these new patterns themselves the focus of attention for the conversations. This is quite different from holding meetings that are "about process."
- ***How* do people interact in the conversation settings?** Old interaction patterns can be changed by focusing on inquiry rather than or as well as discussion and debate. This can include reflection on past experience, discovery of the rationale behind stated positions, etc.

While changing one element (the who, what, or how) may "loosen" things sufficiently to allow new conversations to take place, we believe that the best chances for significant new learning occur when all three are changed.

The primary focus of this document is to suggest techniques for changing the "how" element of conversations. The "how" may be the most essential, although it need not always be changed by design or intention. Certain conditions may allow people to spontaneously discover new ways of interacting.

The "who" and the "what" are addressed to some extent in this document, but other tools and techniques focus more directly on them. For example, two ReuseWorks technologies, the CFRP and ODM (see Section 1.4) are directly applicable to the "who" and the "what," respectively. The CFRP offers a model for reuse interactions that can help in determining who should be involved

in certain conversations. The ODM domain selection and scoping processes provide a framework for negotiating the scope of reuse-specific interactions. Further discussion of the CFRP and ODM in the LIBRA context is included in Sections 5 and 6.

The How: Facilitating New Interactions. Establishing new reuse-supportive interactions is difficult because it requires skills that are not usually associated with software engineering responsibilities. Such interactions require a level of team rapport and communication that is difficult to establish even in conventional project teams, much less in groups of people who may not routinely deal with each other in their normal work processes.

New interactions also require careful attention to the interaction process itself. This kind of thing is usually made easier by the presence of a trained facilitator or process consultant. Yet in the kinds of meetings we envision for users of this document, involving outside consultants or facilitators may be difficult or premature (e.g., an initial meeting when people within an organization first become interested in reuse). At the pre-commitment stage, bringing in an outside facilitator might create rather than remove barriers. This places an increased burden on the skills of internal change agents who must design, coordinate, and facilitate the meetings.

New interactions require shifts in well-ingrained patterns of behavior. When people meet in familiar settings and group configurations, discuss familiar topics, or follow established meeting processes and styles, old behavior patterns are reinforced. In order to create possibilities for change, it can be helpful to design in conditions that "shake up" or disrupt old interaction patterns.

Sections 3.1 through 3.4 below provide an overview of four conceptual tools that are particularly useful for assessing current reuse practice and facilitating new interactions that lead to systematic reuse. The tools are:

- Dramatic scenarios
- System diagrams
- Belief maps
- The Ladder of Inquiry

Section 3.5 discusses how the tools can be used in concert. The References and Recommended Reading section at the end of this document provides pointers to resources for learning more about these tools.

3.1 Dramatic Scenarios

The primary assessment tool in LIBRA is the dramatic scenario. We use the term "dramatic scenario" (or "scenario" for convenience) to refer to a narrative description of events in a given organizational setting. The scenario includes descriptions of the organizational environment and broad business picture, character profiles for the individuals appearing in the scenario, and a sequence of events that focuses more on drama (conflict, tension, transformations) than on task descriptions or workflow analysis.

An Example Scenario

Consider the following scenario:

A software engineer, Claudia, is given an intensive programming task in a specialized area

(domain X) as part of a larger system development effort. After successful completion of the task, Claudia gains a local reputation (within her group) as the domain X "expert." As more related tasks of this kind come up, they are directed her way; with each task, her expertise grows, and the proportion of her time dedicated to this domain also increases.

Eventually, as the learning curve levels off, the tasks become less and less interesting, more and more routine for Claudia. She asks her manager, Tom, for other work but in the meantime domain X has become of critical importance on the critical path for several high-profile projects. Claudia starts feeling "pigeonholed" and requests a trainee so that she can begin to transfer her expertise in domain X and move on to something else. Tom answers, "We can't spare your time training someone right now — you're on the critical path. And we're understaffed as it is."

Elements of Scenarios

- *Context*: An organizational setting is established for the scenario. Ideally, the business environment and other elements of the context shared by the characters in the scenario are presented explicitly.
- *Characters*: Each character in the scenario is described via a character profile that details as much as is relevant about their past experience, life and career situation, values and attitudes towards work, colleagues, etc.
- *The Story*: The main body of the scenario is a narrative describing the sequence of events, interactions, decisions etc. in the story. The story can be told with a variety of alternating forms, including narrative, dialogue and script, or even sample artifacts (email message text, corporate memos, etc.)
- *Key Events*: As a starting basis for analysis and interpretation of the scenario, the story can be annotated to identify key events that warrant focused discussion and commentary.

Why Use Scenarios?

The scenario approach presented here is adapted to the purpose of reuse assessment. Similar techniques could be used for data-gathering as part of a full-fledged domain engineering project, or used in a variety of design or requirements elicitation tasks.

A primary difficulty in reuse assessment is in describing current processes and practices. Reuse is not an explicitly supported process in most development environments. If you ask direct questions such as: "How much reuse happens in your organization?" or "Do you have metrics in place for levels of reuse?" you may be asking people to abstract a level of practice that exists but is not codified in supported processes and policies. If the processes are already codified, they may know how to answer these questions; if not, they may not even understand the question.

Formal process models are good ways to codify processes that we want to institutionalize, manage and make repeatable. They incur a lot of overhead when used to describe and diagnose interactions that we may want to change. Broken or non-optimal processes often have dropped, missing, or conflicting information. To diagnose and fix these problems we need a simple language for describing what the problems look like. Scenarios allow people to identify patterns of interaction and link them to specific experiences in the organization, so that opportunities for systematic reuse can be recognized and understood in concrete, practical terms. These features make scenarios useful tools for learning-based inquiry into reuse.

Features of Scenarios

Some features that distinguish a scenario from a more formal process description include the following:

- *The scenario tells a story.* It need not be factual (i.e., not a specific documented case study); if factual, the specific organization/people need not be named. On the other hand, the story is narrated with respect to a particular character, not a formalized role. Much of the value of the scenario as a starting point for interpretation stems from this quality of concreteness. Because the scenario deals with a "story" it engages the listener's imagination without raising issues of historical correctness ("it didn't happen that way..."). But the presence of specific data about the characters makes the description richer than an abstracted process. It is easier for people to see themselves in the story, to relate the story to incidents in their own experience, and to see the multiple possibilities for action in the story based on the characters' beliefs and choices.

- *The scenario's story has a beginning, a middle, and an end.* By contrast, a process model describes a set of interactions that are repeatable (an input-output flow). Since process models are used to describe workflow, the statement that there is an "established process in place" implies that the organization can respond to ongoing inputs of the same type and produce similar outputs. This applies best to a classic production environment, somewhat less fully to a software development environment where each system developed may present unique design issues; and even less well to knowledge-intensive activities.

  The beginning and end of the scenario are somewhat arbitrary: the "frame of the picture." There is a sense that events have led up to the scenario and that other events will follow afterward. The frame of the scenario is determined by the scenario author in order to place central emphasis on certain significant events. Thus a single scenario can be discussed in its entirety within a single meeting; it is not attempting to provide a complete, seamless description of process.

- *The scenario often describes a situation with inherent instability.* Action or events in a scenario can be the actions or events that occur when ordinary business processes break down or are in transition. Where repetition does occur in a scenario it may imply a problem, e.g., a kind of "stuckness" in the situation.

  By contrast, formal process models usually are valid only to the extent that the situation described stays relatively stable. In fact, they are generally meant to enforce stability (recent work on "evolvable" process models notwithstanding). In principle, process modeling could be used to describe change processes in organizations. However, historically they have not been used in this way, probably in part because such processes are not usually very orderly or repeatable.

- *The scenario can collapse and expand time-scale to support analytical purposes.* Scenarios can alternate between fine-grained descriptions of interactions (even scripts of conversations) with broad-brush narrative that covers large-scale events ("Over the next few years Claudia worked on several more projects...") The scenario can help make the rhythm of unfolding events more discernible: e.g., an acceleration of momentum, hurry-up-and-wait dynamics, etc.

  This can provide visibility into the interactions between short and long time-scale processes. For example, a scenario can describe the dynamics whereby apparently "stuck" situations erode and finally change. Scenarios can describe the role of learning in an organization. Multiple iterations through a knowledge-intensive process change the state of the situation — the participants learn more, or the overall business situation shifts. Yet this learning, since it is

not a tangible "output," cannot be easily expressed using typical process modeling approaches.

3.2 System Diagrams

System diagrams are pictures describing human interaction patterns in organizations. They include positions, actions, and impacts on other players. Most often they are used to show a breakdown or vicious cycle of self-defeating actions. They can also be used to show effective patterns of action that are adaptive to change and self-correcting.

Working from the example scenario in the previous sub-section, we present a small example of a system diagram that interprets the interactions described in the scenario. Here is an interpretation of the first part of the scenario:

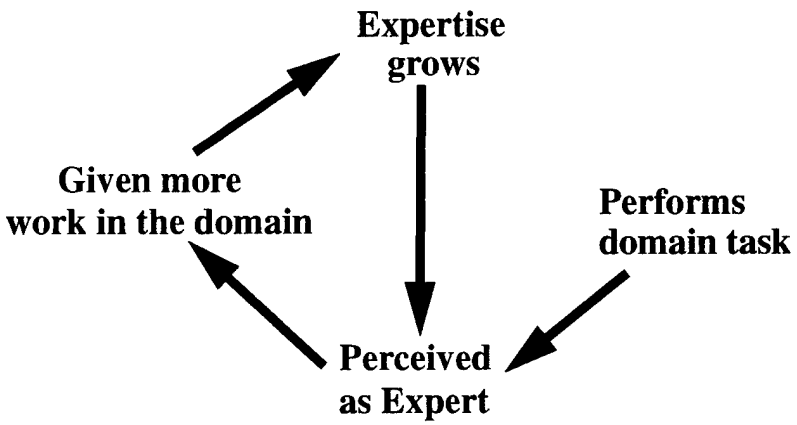


Exhibit 2. Example System Diagram

The scenario shows the cause-effect connection between events in the scenario: Successful completion of the project task leads to increased perception of Claudia's expertise, which leads in turn to more related work being assigned, etc.

A few points to note about the system diagram notation:

- The diagrams are informal. They are most useful for initiating dialogue and discussion; rigorous formality would impose a learning curve that would reduce their usefulness. There is no one way to draw them, as exemplified by the two different approaches we use in this document.
- The diagrams show *patterns* of events, trends, and interactions over time. They are not designed to show the flow of input and output data in a network of processes.
- The diagrams become most interesting when structures like flip-flops, cycles, and feedback loops (positive and negative) appear. (The diagrams are sometimes annotated with see-saws and spinning circles to make these structures more evident.) The simple example of Exhibit 2 shows a positive reinforcement cycle at work. The more assignments in the domain Claudia receives, the more her expertise increases and the more work of this kind she is assigned.

One of the insights encouraged by system diagrams is that every action has a reaction; in particular, positive feedback loops within one time window may evoke counter-balancing forces within a

larger time window. In the next diagram, we see the effects of "too much of a good thing": after a while, what was a chance to learn and gain new skills becomes routine and even boring. Claudia is ready to move on to other things. Unfortunately, management (in this scenario) may come to depend on her skills in this one area and begin to box her into the "resident expert" role. Another natural response to gaining a certain degree of expertise is the wish to codify that expertise or pass it on to others. Again, the climate of the organization may be such that "no one can be spared right now" to take on this "apprentice" task. This is illustrated by two new loops in the diagram in Exhibit 3, in this case "negative feedback" loops (in terms of the frustrated objectives of at least some of the players involved).

Negative feedback loops ("vicious circles") of this kind are one way that situations get "stuck." System diagrams help make the dynamics behind such situations more visible in several ways:

- They show patterns of interactions that may be invisible to any one player in the scenario. Each of us tends to view our interactions with others from our own perspective. Through the process of building a common system diagram, we can see how our perspective interlocks with the perspectives of others. (The exercise can be even more valuable if we are brought together with people with whom we would otherwise not interact.)
- They can "telescope" interactions that usually take place over longer periods of time into patterns that we can take in as a whole. In this example, Tom, the manager who frustrates Claudia's repeated requests for new types of assignments, may see only the time-crunch of the next critical project, and not look far enough ahead to realize that Claudia may quit.
- They can reveal how actions and strategies may work at cross-purposes to their (apparent) intent, in part through the shifts in scope, perspective, and time-frame explained above. In the example scenario, Tom's policy was based on the perceived critical importance of continued access to Claudia's expertise. However, if pushed too far, the policy may produce just the opposite result: i.e., it may cost the company access to Claudia's expertise.

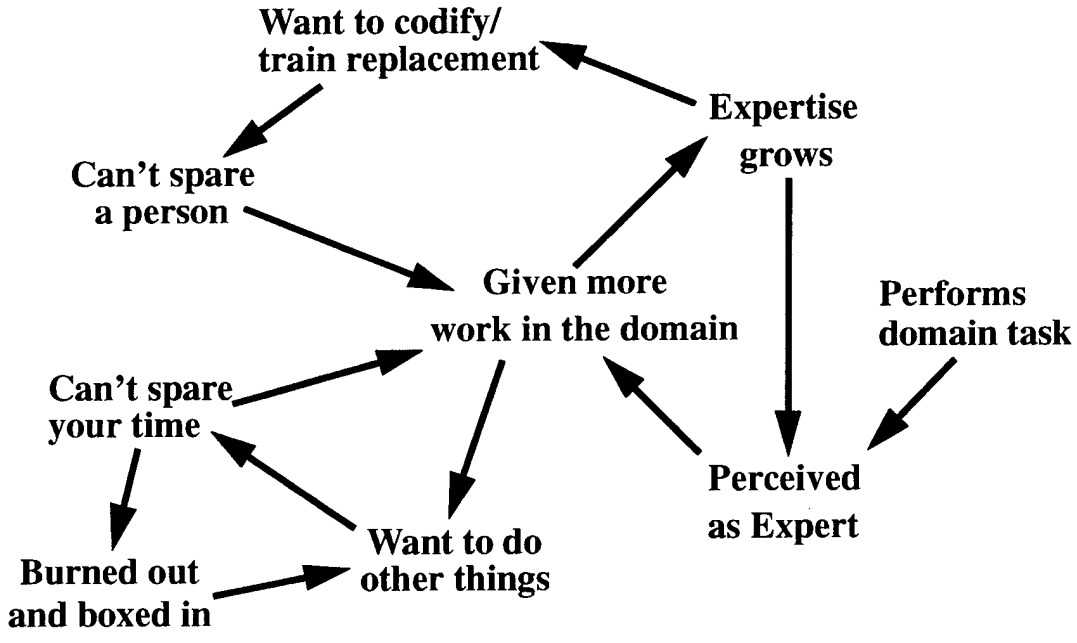


Exhibit 3. Negative feedback loops in System Diagrams

- They make it easier to recognize recurring interaction patterns in different contexts by providing intuitively accessible representations of these patterns. To continue with the example scenario, suppose Claudia announces she is quitting. She is willing to stay on long enough to train a replacement. It might be all too typical in such a situation for Tom to use Claudia's remaining time on that last high-priority project task—thus perpetuating the same cycle that led to her resignation in the first place. (Claudia will probably then be brought back as a consultant...) If Tom saw this pattern enough times, he might begin to learn that the strategy that seems to make the most short-term sense is counter-productive.

System Diagrams Suggest Corrective Actions

So far, the primary benefit of system diagrams may appear to be as an assessment or diagnostic tool. However, if used as a basis for reflection and generating insight, they can reveal possible strategies to get situations "unstuck," or conversely to bring inherently unstable and chaotic situations back towards healthy equilibrium. Quite often, interpretation of a system diagram may reveal an effective course of action, or leverage point, that is counter-intuitive to the players involved. (A good example in the software arena might be Sun Microsystem's decision to develop a public standards-based rather than proprietary workstation operating system.)

In the example scenario (admittedly highly simplified, for the sake of illustration) a number of alternative strategies could be suggested. The following are some examples of possible next steps that would change the dynamics of the situation:

- Institute an "apprentice/mentor" program within the company that brings new hires together with developers experienced in particular aspects of the system.
- Create similar "co-mentoring" plans where developers with complementary expertise exchange information in their respective areas.
- If the employee is already leaving, allocate time and resources to debrief the employee thoroughly.
- If the expertise is of strategic, proprietary value to the company, consider giving the expert a budget to codify the knowledge in some form (a training document, a formal domain model), offering as an incentive the chance to move into a new area once the expertise has been demonstrated to be transferable.
- If the expertise is of strategic but non-proprietary value to the company, consider giving the expert opportunities for "micro-entrepreneurship" by selling the expertise outside the organization as consulting (this demands that the knowledge hold up as "best practice" within the marketplace).

Each of these strategies will place different kinds of demands on the individuals' and organization's capacity for change. One set of options involves transferring knowledge from one person to a new person (passing the torch). Another set involves codifying the knowledge so that it becomes more of a corporate asset and less dependent on individual personnel. The last option might require a shift in the business model of the firm. At some level, however, all these options may require a substantive shift in behavior on the part of management: to allocate resources to learning activities when production goals appear to be paramount in the business environment. The CFRP provides guidelines for such activities, which are often not supported explicitly by the organization.

System Diagrams and Scenarios

There may appear to be considerable overlap between system diagrams and scenarios. For example, both help to uncover "blind spots" of local or short-term thinking. However, the techniques are complementary but quite distinct. Scenarios are built from concrete data (either real or hypothetical). The character profiles, for example, are an essential component of scenarios, while they are generally lacking in system diagrams.

In principle, one could model an entire scenario using system diagrams as a kind of informal process modeling, although this is not the strength of the technique. Instead, system diagrams are most useful in interpreting scenarios in terms of their key events and the inter-relationships between those events. Learning occurs when a scenario is interpreted via system diagrams, or conversely, when a given diagram's pattern is recognized when reading a scenario.

System Diagrams as Archetypal Patterns

In general, a dramatic scenario tells a story that includes several instances of *archetypal* interaction patterns that can be represented by system diagrams. How does one recognize the scenes or situations in the scenario that are most usefully described using system diagrams? What lends a given set of interactions a quality that might be termed "archetypal"?

- *Structure*: The pattern of interactions has a clear form, topology, or other formal property such as symmetry, that creates a "signature" recognizable in a variety of situations.
- *Coherence*: The pattern has cohesion and independence; it can be isolated from the surrounding network of other interactions so that a small set of interactions become the focus.
- *Resonance*: The pattern resonates with observers; that is, they recognize it as a recurring pattern from their own experience in several different environments.
- *Reinforcement*: The pattern tends to intensify the energy of its constituent interactions; through strong forces and counter-forces the pattern itself can have an impact on interactions disproportionate to the importance of the interaction content (e.g., escalation of an argument about a minor issue).
- *Significance*: The process of observing the pattern in its diagrammatic form reveals some significant insight about the nature of the interactions. This meaning can be thought of as the "moral of the story."

The insight gained from inspecting an archetypal system diagram can be of particular value when it is counter-intuitive to the characters' beliefs. This may involve the discovery of ***self-defeating*** patterns: current behaviors that produce results contrary to the intended outcome.

For example, someone setting up a reuse library may try to create incentives for reuse by rewarding engineers who place components in the library. The intent of the reward system is clearly to encourage design for reuse. But if the incentive program is not handled carefully, it may create incentive to submit lower-quality components, to overcrowd the component base with close variants, or to actually discourage use of already existing components.

In combination with more explicit models of the beliefs of the key players, system diagrams can be pushed beyond description of merely self-defeating activities, to include patterns of interaction that might be termed ***self-sealing***. Persons in an organization may engage in self-defeating behavior and yet still be able to be confronted with the effects of this behavior and correct it. Self-seal-

ing patterns occur when there are strong mechanisms in place to dismiss or resist diagnoses of the connection between actions and results.

If it were not for self-defeating and self-sealing interaction patterns, system diagrams alone might be sufficient to allow people to discover new options for organizational improvement. However, sometimes options remain hidden because they require shifts to pervasive, underlying beliefs held by players in the scenario. Since system diagrams do not represent these beliefs explicitly, the beliefs may not surface directly in dialogue about these diagrams. To discover new options based on shifts of beliefs, a deeper level of analysis may be required. In this document, we refer to this approach as "belief mapping," as discussed in the following section.

3.3 Belief Maps

Some Example Beliefs

Returning to the example scenario introduced earlier, it is possible to describe some underlying beliefs of the players. The main dynamic (the "drama" as it were) lies in the tension between the beliefs of Claudia and Tom. Claudia's belief could be stated as follows:

> "My value to this organization goes beyond the specific knowledge I have acquired in Domain X. I should be valued for my ability to rapidly acquire knowledge in new areas, to transfer knowledge from one area to another, and to create and codify new knowledge in a form that is transferable to others. Dealing with me this way is not only the best strategy for me to meet my individual objectives for career and technical growth, but is also the best overall strategy for the organization."

By contrast, Tom appears to be operating from a belief of this sort:

> "Training people in new areas is expensive for the company, whereas putting that expertise to work returns value for the company's investment. Once an employee has acquired adequate knowledge in a technical area of critical importance to the company, it is not in the company's best interest to dilute that person's focus onto other areas."

While drastically simplified, these representative belief statements highlight some important aspects of belief mapping:

- Individuals hold multiple, related beliefs. These separate assertions can be thought of as woven into a "personal belief map." This personal belief map can include conflicting or even contradictory beliefs. The process of personal reflection or dialogue with others can bring such contradictory beliefs to attention. This can be a powerful experience that leads an individual to shift a belief (without having been persuaded to adopt a different belief through debate or argument).

- Belief maps can be used to show relations between beliefs of multiple players. Often, the beliefs of different players in a scenario may conflict; they may be aware or unaware of the fact that they are operating out of different beliefs or assumptions. Such dissonance in beliefs typically engenders conflict in discussions. Making conflicting beliefs visible to each player, without simultaneously trying to resolve or reconcile them, can be a powerful inquiry technique.

- Organizations as a whole may also be thought of as embodying sets of beliefs. These are

often tacitly shared by members and thus form part of the implicit culture of the organization. However, individual beliefs may also be in conflict with the organization's beliefs.

In general, to fully understand the dynamics of a given situation we need to know the beliefs from which each participant is operating. This illustrates a key value of dramatic scenario interpretation, with its emphasis on fully "characterized" characters as distinct from the more generic roles of process models. A complete scenario description includes up-front articulation of each player's core beliefs. During scenario interpretation and discussion, participants can test whether scripted interactions "ring true" for the characters as described. They can explore what data in the interaction would hide or reveal the underlying beliefs to the other players. In assessing real organizational situations, on the other hand, we must reason backward from the data of people's actions, decisions and stated positions; and we must do this reasoning in situations where we are involved as stakeholders and constrained by our own active beliefs and assumptions. Scenarios provide a way to practice these interpretive skills before attempting this difficult feat of "reverse engineering."

Elements of Belief Maps

A belief map can be represented as a diagram where "nodes" are short-hand phrases that stand for beliefs and various kinds of arcs connect these nodes to represent relationships between beliefs.

- An individual's beliefs can shift over time; similarly, the common beliefs that shape an organization's overall pattern of behavior can undergo broad shifts.
- A community is defined not only by the common beliefs of by its members, but by controversies, schisms or polarities that divide and structure the community.
- The contrast between beliefs can be illustrated effectively through a description of an individual's shift of belief over time, or through dramatic conflict between players operating on the basis of different beliefs.

Key relationships include:

- Inference: On the basis of holding belief A, the person infers belief B as well.
- Reinforcement: Two beliefs seem to imply each other. Questioning either belief might call the other into question.
- Conflict: Two or more beliefs appear mutually exclusive. If one person holds both beliefs then usually one of the beliefs will be undiscussable or relatively hidden to that person. If the conflict occurs between people it may signify an occasion for argument or competition.

Beliefs and Alternative Scenarios

To illustrate the impact of the underlying belief maps on scenario outcomes and system diagrams, let us return to the example scenario but replace Claudia with Howard, an engineer who is a few years away from retirement age. Like Claudia, he has acquired expertise in an essential sub-system area of the organization's main product. Howard's operating belief may be along these lines:

> "My knowledge of this system area is my sole remaining source of job security here. The company could not get along without my expertise, because it would take a long time for anyone else to come up to speed on this part of the system."

This can lead to what might be termed the "knowledge hoarder" syndrome: an engineer who tries to become and remain an indispensable source of information. This can be problematic, particularly for a manager who values continual employee learning and knowledge sharing. In such a scenario, pressure to move Howard to a different technical area or to train a replacement is more likely to be initiated by the manager and resisted by Howard. The final outcome may be "firing the deadwood." Such a situation is almost the mirror image of the "expert burn-out" syndrome in the scenario with Claudia as protagonist.

On the other hand, if Howard's manager is operating out of a similar belief framework, there may be little open conflict in the situation. However, this can lead to an organizational climate that fosters "guarding of turf," strong divisional barriers, and a tendency towards complacency. Both individual engineers and the company as a whole may cling to competencies that are core ("that's what we do!") but are no longer strategic (no one cares anymore) or competitive (others are doing it better).

Such alternative scenarios suggest the importance of factors such as:

- The interaction of different players' beliefs (e.g., Claudia or Howard and their managers);
- The relation between players' beliefs and the overall beliefs and norms of the organization;
- The viability of organizational beliefs within the current business environment (e.g., the strategic importance of Domain X).

Belief mapping is an important technique for the reuse inquirer. Typical technology transfer implies education in a vacuum; that is, since new technology is being introduced the assumption is that members of the adopting organization will be unfamiliar with the technology involved. However, because of the close linkage between general reuse concepts and people's specific software practices, engineers often think they understand reuse from their own experience. This means that reuse adoption efforts are interpreted within an engineering culture that has its own beliefs about reuse — what works and doesn't.

Mapping these beliefs is useful in several ways. At a minimum, it can help the reuse inquirer better understand where the points of reuse receptiveness and resistance lie. In addition, it can help sensitize the inquirer to the reasons why certain beliefs exist in a given setting. For example, previous experiences may have helped to confirm beliefs or expectations among the players. Or certain beliefs may help create a stronger sense of community within the organization. Or a belief may be a reaction to certain harsh realities in the environment. By inquiring about the evidence behind given beliefs, reuse inquirers can avoid falling unintentionally into argumentation or persuasion. If belief mapping is conducted collaboratively among the players, the activity itself may create opportunities for constructive dialogue.

System Diagrams and Beliefs

To clarify the relationship between system diagrams and beliefs, consider that a system diagram represents players' *actions* and *positions* in a given situation. We use the term "position" to mean decisions or stated preferences about actions and outcomes; thus a position could be stated in the form, "I intend to do X," or "I think we (the company, design team, etc.) should do X in this situation." In contrast, a belief is an assumption about "how the world is" that is part of the reasoning behind actions and positions. The dynamics represented in system diagrams are thus fueled by the underlying beliefs of participants. A system diagram by itself does not always illustrate what keeps the dynamic persistent over time (the stability of desired behavior or the stuckness of undesired behavior). For example, a boom-and-bust cycle may persist at a company because managers believe they have no sensible option but to overstaff at peak times and lay off in lean times.

Related Work

Belief mapping is the most exploratory element of the approach presented in this document and the least thoroughly validated by experience in other fields. It is closely related to work in the learning organization field on *mental models*, maps of the reasoning behind people's behavior and decisions that are largely inaccessible to conscious reflection {Senge90]. Another body of work, focusing on deep-seated cultural and social norms within organizations (e.g., [Sche85, Sche93]), is applicable to beliefs specific to software engineering, reuse, and knowledge creation. Chris Argyris has proposed a theoretical framework that distinguishes between people's espoused theories (what they *say* are the reasons for what they do) and their "theories-in-use" (the beliefs out of which they operate, based on observations of their actions). Our belief mapping approach borrows from these ideas and integrates them with cognitive and interactional domain modeling skills, as exemplified in the Organization Domain Modeling (ODM) method.

3.4 The Ladder of Inquiry

In the example scenario described above, Claudia's belief is characteristic of a knowledge-creating organization — if, that is, the belief were shared by her manager and the organization as a whole. We have described alternative scenarios where the "block" was in the beliefs held by the engineer (Howard), the manager (Tom), or the organization as a whole. This raises the question: how does change happen in such situations? How can shifts occur in the underlying belief structures, allowing movement toward the dynamics of a knowledge-creating organization?

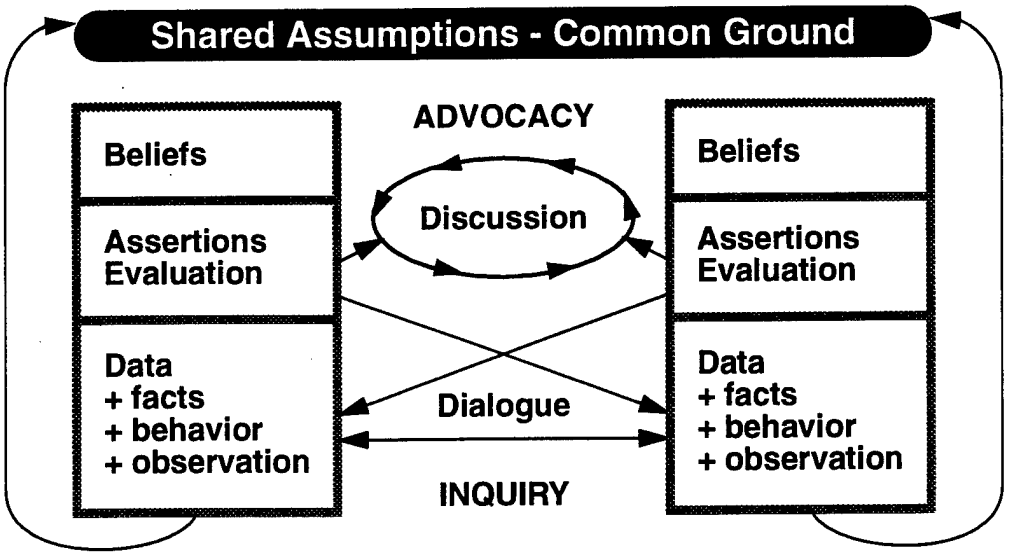
Shifting Beliefs. A fundamental assumption of the LIBRA approach is that people cannot be coerced to shift their beliefs. Shifts in belief occur from within, when people conclude on their own that their prior beliefs no longer match their reality. The tools discussed so far — dramatic scenarios, system diagrams, and belief maps — can help people to describe and analyze their reality in a way that can lead to such shifts in belief. One of the key problems in using the tools for this purpose is that it can be very difficult to discover the beliefs which underlie the positions and interactions described in system diagrams. People usually do not articulate their beliefs freely; often they hide their beliefs or are not even consciously aware of them.

There are inquiry techniques that can be practiced in direct face-to-face conversations that can help to address these issues and increase people's receptiveness to shifts in beliefs. In this section, we introduce a tool, the ***Ladder of Inquiry***, for diagnosing and enhancing the quality of inquiry. The Ladder of Inquiry aids in understanding how the other tools operate and why they are effective.

Conversation, Discussion and Dialogue

The focus of the tools up to this point has been on general ***interactions***, exchanges of information, promises, requests, or assertions between people. We will use the term ***conversation*** to denote a specific face-to-face verbal interaction between two or more people. Direct conversations are the easiest settings in which to recognize the dynamics of advocacy or inquiry. Depending on the dynamic or quality of a conversation, we refer to it as ***discussion*** (more advocacy) or ***dialogue*** (more inquiry).

When we think informally about the dynamics of a conversation where participants begin with different positions, our usual picture is a borrowed mechanical metaphor: positions collide with each other until one party successfully persuades the other to capitulate or shift position, or until some compromise is reached. (David Bohm cites the relationship of the word "discussion" to "percussion," literally "hitting against each other." [Bohm90]) This shift may be conceptual (we



Source: Adapted by Larry Levine from C. Argyris

Exhibit 4. The Ladder of Inquiry

are convinced by the rational arguments of the other) or based on power and authority (one party ultimately has the power to decide). We will refer here to this conversational dynamic as *discussion*. In discussion, one's primary attention is focused on stating one's position, and meeting disagreement with argument and persuasion. Because people are discussing issues that are important, they are highly involved and interested in the outcome. Because the outcome will result in "winners" and "losers" the discussions can generate varying degrees of emotional "heat." In contrast, *dialogue* implies a willingness to explore the rationale, beliefs and experiences underlying other people's positions.

The Ladder Metaphor

The Ladder of Inquiry (see Exhibit 4) provides a framework for understanding when conversing parties are stuck, competing, sparring, making discoveries, clarifying decisions, or misinterpreting each other's meaning and intent. The ladder metaphor illustrates choice of movement in conversation, with one direction (down the ladder) leading toward collaborative discovery and the other (up the ladder) toward competitive debate. Both are needed, and distinguishing them clearly often makes both more effective.

The rungs of the ladder reflect the qualities of listening, receptiveness, and resistance (from bottom to top). When conversations move up the ladder, assertions are defended by appealing to the beliefs that support them. This often leads to debate in which differing parties seek winning over discovering. Assertions and beliefs are pitted against one another in the form of advocacy or discussion. Moving "down the ladder" means using inquiry techniques to uncover the reasoning and assumptions (the "data") behind a given assertion or position, engendering more opportunities for dialogue.

Falling Up the Ladder. There is one sense in which the ladder metaphor can create some confusion. In general, unless we are careful and disciplined in our conversations, the tendency is either to move unconsciously up the ladder or to stay stuck at one level in an unproductive way. On the other hand, it usually takes some self-restraint and personal discipline to intentionally move down the ladder. This means, in effect, that gravity makes us *fall* "up the ladder," whereas we must *climb* by dint of personal effort "down the ladder". We leave the metaphor in its current form

because the notion of moving "up" the ladder corresponds closely to the physical sensations and intuitions we experience. Conversely, to move down the ladder we must generally "damp down" our immediate emotional reactions.

In its simplest form the Ladder can be used as a diagnostic aid to assess how a given conversation is flowing: Are positions hardening? Are people able to inquire beneath the initial positions against which they are tempted to react? The Ladder can also be used as a guide to monitoring one's own behavior within a conversation. The scenario analysis in Section 4 illustrates how the Ladder of Inquiry can be used to diagnose breakdowns in communications or characterize alternative paths that could have been followed.

In assessing or guiding conversations using the Ladder, a few guidelines can be applied:

- The most productive conversations involve intentional moves on the Ladder. That is, there are times when it is appropriate to move up the Ladder and close on decisions and actions; there are also times when the ability to move down the Ladder is critical to success.
- Staying at one level of the Ladder may require as much discipline as shifting levels intentionally. Discussion can involve an inadvertent tendency to drift up the Ladder.
- People also frequently get "stuck" at particular rungs of the Ladder. To the extent that conversations stay at the surface level of conflicting positions, and do not move downward to discovery of underlying beliefs, conversations can break down, move in circles, oscillate back and forth between alternatives, or reach apparent resolutions that dissolve afterward.

There are a variety of techniques for addressing "stuck" patterns and allowing discussions to get unstuck. These include techniques for moving up or down the Ladder as well as for making more productive lateral moves.

Relationship of the Ladder of Inquiry to Other Tools

Scenarios and system diagrams describe behaviors. The *content* of scenarios and system diagrams can include interactions that move up or down the Ladder of Inquiry. Moving "up the Ladder" often leads to increased and more strident advocacy (leading eventually to some sort of process breakdown); moving "down the Ladder" often involves interpretation of the underlying rationale behind positions.

Belief maps lie further down the Ladder than scenarios or system diagrams. They can be defined at various levels: stated positions or alternatives, explicit assertions, or more hidden underlying beliefs.

While scenarios, system diagrams, and belief maps each live at different rungs of the Ladder of Inquiry, they are similar in that each involves *staying at the same rung of the Ladder.* In contrast, the Ladder of Inquiry technique itself concerns ways to make intentional shifts between rungs of the Ladder.

The common element in these techniques is to make a shift in *process* rather than a direct shift in any position. The changes involve deliberate shifts of focus:

- From decisions ("We shall do this...") to recommendations ("We should do this..."), to possibilities ("We could do this...");
- From search for a single-point outcome to enumerating a set of possible outcomes and their relationships (e.g., from a *point* solution to a solution *space*);

- From positions to the beliefs and rationale underlying the positions.

3.5 Applying the Tools

To use the LIBRA tools most effectively, it should be possible to move from scenarios to system diagrams and from scenarios to belief maps, as part of reflection and interpretation. Although we do not expect that scenarios will be appropriate for all organizations, or for all situations within a given organization, scenarios play a central role in LIBRA for several reasons:

- Scenarios offer opportunities to apply other inquiry techniques as part of interpretation and analysis. Scenarios thus serve as an integrating framework for inquiry-based self-assessment.
- Scenarios transfer well from an individual's directed reading, to written commentary and discussion within a group, to interpretation in meetings.

Because inquiry-based self-assessment involves the interpretation of scenarios that describe situations distinct from the real organizational setting, the scenario approach gives people a chance to practice their inquiry skills on neutral ground. Dramatic scenarios can be about hypothetical or real case studies. One can practice the skill of recognizing and diagramming a system archetype in the scenario without having to maintain the distance that is often needed to recognize an archetype in one's current situation.

The easiest way to use scenarios is to convene a group of people to read (usually prior to the meeting) and discuss an existing scenario, such as the one provided in Section 4. The scenario can be interpreted and analyzed using a combination of free, unstructured dialogue and the other tools for inquiry provided in this document. Here are some possible strategies for using the tools together in analysis:

- Starting from the scenario, identify what appear to be significant or key events in the story. Try to transcribe the events in terms of a system diagram or belief map. When applied successfully, a pattern will be discovered.
- Start from a known set (or casebook) of system diagrams or belief maps that point to key issues encountered frequently in reuse. Scan the scenario for instances where one of these archetypal patterns seems active; then sketch a diagram to describe how that archetype is played out at that point in the story.
- Work from specific scenarios to the underlying belief maps of the characters involved, using the Ladder of Inquiry.

It may appear counter-intuitive to describe this process as a form of "assessment", since there is nothing in the process that directly involves compiling and validating data about the organization. A group of people could wind up discussing a "story" about another hypothetical organization. This would seem to have little to do with assessment in a conventional sense.

Nevertheless, we believe this technique can provide a reasonable starting point for an organization's reuse self-assessment. Scenario interpretation provides a scaffold for the ensemble of inquiry techniques presented in this document. The very indirectness of the scenario relative to the work practices and business environment of the organization can be an important benefit of the technique. Because the scenario does not directly refer to events and decisions that affect the participants as direct stakeholders, the interpretation can take place on relatively neutral territory.

4.0 Reuse Adoption Scenario and Analysis

In this section we present a fictional scenario that identifies many of the key patterns and choices impacting reuse adoption (for better or worse). Our intention in using this "case study" format is to characterize the patterns and choices in a very specific, concrete form. The case study gets to the heart of how stakeholders interact in a business environment, and what the real obstacles to reuse are.

Our approach uses the metaphor of theater. The case study depicts some of the key "scenes" in the "drama" of reuse planning. We present the drama in terms of its characters, their relationships in the organization, and a succession of scenes showing one path towards institutionalized reuse.

Key interactions between characters — those indicating significant risks or choices being made — are flagged with a number surrounded by angle brackets in boldface (e.g, **<1>**). For each of these key events in the scenario, we provide commentary (numbered accordingly) in Section 4.2 ("Scenario Analysis"). In these commentaries, we analyze and interpret the events, often by highlighting the underlying dynamics through system diagrams and by suggesting tools that can be used to steer such turning points towards a successful outcome.

4.1 The Scenario

First, some context about the setting in which the story occurs:

The Organization

The case study concerns a Division within a systems development firm that contracts with both commercial clients and the Federal Government. The Division consists of approximately 400 staff. Reporting to the Division Head are the following people:

- Business Development Manager
- Director of Technology
- Director of Engineering
- Line of Business Managers

The case study focuses on one Line of Business (LoB) with a staff of about 80 employees. At any given time there are between three and ten active projects in this LoB, each headed by a Project Manager.

Another important part of the organization is a corporate-wide software technology transfer group, which is chartered with bringing the latest and greatest methods into the operating divisions.

The Players

The key characters in this study are the following:

Joe: A senior software engineer

Mike: A more junior engineer

Janice: A software reuse guru reporting to the Director of Technology

Jim: Project manager to whom Joe and Mike report

Andy: Marketing person reporting to the Business Development Manager

June: A senior software engineer working on a different project in the same LoB

Ross: The Line of Business Manager.

Because much of our understanding of the case study will derive from an analysis of the characters' beliefs and motivations, we present a brief profile of the characters, describing their goals, expectations, pleasures, and frustrations on the job.

Joe

Joe, age 40, is a Senior Software Engineer, a closet believer in reuse who is cynical because of his experience — he has seen the snake oil. His job, as he sees it, is to do excellent technical work on time and within budget, to provide good documentation, to teach more junior members and serve as a model for them, to give his opinions when asked by higher-ups, and to support the directions set by those above him. What he requires to do these things are, primarily, clear specifications, time uninterrupted by meetings, adequate hardware, and some contact with those who are still idealists and can keep him from becoming totally cynical.

Joe expects those above him in the corporate hierarchy to stay out of his way. He expects the customer to love everything he does. He expects his colleagues to be smart and honest; and he expects marketing people to talk straight and respect his art.

Joe is happy when the software works, when the problems that arise are those he anticipated, when he can say "I was right," and when his technical decisions stand the real-world test. He dislikes politics, fads, psychologizing, and customer arrogance. Joe is not quite up on new technologies and technical trends. He often dismisses them as "the same old thing." But he is open and is interested in reuse. He prides himself as a good reuser and designer of reusable code. He is semi-convinced that systematic reuse is better than the ad hoc variety.

Mike

Mike, age 27, is a relatively junior software engineer but he has valuable experience in a variety of application development contexts, as well as some academic grounding in cognitive aspects of software engineering (he recently obtained his Master's degree in Cognitive Science). His job, as he sees it, is to complete his development assignments on schedule with reliable, understandable, and maintainable code. To do this, he needs clear and stable specifications, and a decent software development environment: efficient compilers and linkers, a rich set of design, programming and debugging tools, and fast Internet access so that he can post technical queries and stay abreast of what the rest of the industry is doing.

Mike finds his current environment moderately useful, but limited because the tools are not fully integrated; he knows that a lot more is possible. He is also frustrated by the unstable infrastructure configuration, which frequently causes tools to disappear from where he expected to find them and, after much wasted time searching, reappear on another system or in a different directory.

Mike is a bridge builder: he enjoys informal technical talk with his peers, which he believes is the basis for growing a common development culture. He expects his colleagues to be open in sharing

techniques, approaches, components, and experiences; he expects them to be willing to take the time to discuss design tradeoffs and, through such exchanges, to build the common culture. He views this process as the essence of high-powered software development groups.

Mike dislikes isolationism in other developers ("I have my job and you have yours"), and is frustrated when his colleagues and management respond with skepticism to his suggestions for productivity and quality enhancement. He is frustrated by the lack of an overall vision of excellence among the LoB's software staff, and he believes that a lot more would be possible if there were a will to achieve. Mike's motto (which he is careful not to voice to the wrong people) is "Work smart not hard."

Janice

Janice, age 38, is a technologist. Her job is to improve software quality and productivity in the Division. She believes this can best be done by institutionalizing reuse. To accomplish this she needs access to software engineers: a reasonable commitment of their time as well as their cooperation. She would like, but does not have, her own small staff of software engineers dedicated to tool building, tool support, and infrastructure development.

Janice expects to encounter resistance to the changes she proposes, some of it due to inertia, some to lack of understanding. Despite this, she sees an overall willingness in both engineering and management to do whatever will enhance the company's competitiveness — if they can be convinced of this outcome.

Janice is thrilled when she sees process improvements take hold and spread through the Division. She dislikes pessimism and stationary inertia, the tendency to do "what we have always done" in the belief that it cannot be any other way.

Janice has the ear of the Directory of Technology, who worked with her in a previous company and trusts her insights and judgments. She has been empowered to spend time working with LoB projects in an attempt to get products out the door "better, faster, cheaper." She has been given a small amount of additional funding to participate in reuse and software workshops, including writing and submitting her own papers.

Janice sees lots of opportunities for making real change happen, and this is what motivates her. Over the years, however, she has seen process improvement and technology efforts come and go, and she is getting a little tired of her Division's tendency to shift direction radically upon the first indication of a problem. She is always saying to herself and others, "We can really make this happen if we just stick with it."

Jim

Jim, age 42, is a software project manager. His job is to ensure that products are developed and delivered on time and within budget. To do this, he needs several things: clear product requirements specifications, advice from his engineering staff about technical issues and alternative solutions, and timely information about the cost, schedule, and technical status of his project. In particular, he needs to know quickly about any problems appearing on the horizon.

Jim fosters an environment in which pride is shared by all when project challenges are met. He expects his staff to take ownership of their tasks, to be committed to getting the job done, and to be willing to change approach or shift direction if it is necessary for the benefit of the project. He manages a good mix of junior and senior software engineers including some fine, trustworthy people; he also has some less than stellar performers whom he tries to pair up with his best people

in a mentor-advisee relationship. Sometimes this works and sometimes it doesn't, but Jim has no illusions of ever having a team of only superstars: he is too firm a believer in Murphy's Law.

From his Project Manager colleagues, Jim expects the same kind of burnout that he is experiencing, lots of pleasantries and mutual groaning exchanged between them but little substantive interaction. From management he expects, above all, an interest in customer satisfaction, and a grudging willingness to provide him necessary resources if he fights hard enough for them.

Jim is happy if his customer is happy; he is unhappy when his customer is unhappy: it is really that simple. He always has that look on his face that says, "If it's not about my project, Go Away." He is overworked and underpaid — has been for a long time. Still, he usually gets the job done, customers like what he delivers, and he's still employed: nothing to sneeze at when all is said and done.

Andy

Andy, age 45, is a marketing person. He is a good-natured fellow who has retained remarkable enthusiasm for his many years of experience. His job, as he sees it, is to identify potential customer niches and specific sales opportunities; to cultivate a desire in potential customers for what the LoB has to offer, and convince them that the LoB is the best possible source; and to work with the LoB Manager and project managers to enhance the marketability of their products.

To do this job, Jim needs market data, travel dollars, enough technical information to be credible, and access to engineers for backup presentations. He expects others to provide him with timely information about the LoB's products and potential competition. He expects the LoB's engineers to help in marketing, and he needs funding from upper management to support the development of marketing materials.

Andy is happy when he sees the business base steadily expanding. He becomes unhappy when he feels he has to market a "bill of goods" that has low value. He has a graphic artist on staff to create marketing pamphlets and promotional literature, and an expense budget that he views as insufficient.

Andy gets much of his product information from periodic LoB planning meetings and many ad hoc meetings at all levels of the LoB. He has good contacts among the potential customer base, and he draws strategic insights from his conversations with these folks.

Andy does not fully understand the new software methodologies; however, he appreciates the basic principles enough to believe they have marketing potential. He finds working with bright engineers exciting and fulfilling, and takes satisfaction in being able to convey key ideas to customers who do not speak the technical language. Andy's guiding principle is: explain it to me simply, and I can give your idea a wide audience.

June

June, age 31, is a dyed-in-the-wool software engineer. She is skeptical about managers, who she feels do not know anything about software and always ask for more than is reasonable. She is skeptical about technology "hype" which she feels is a waste of time and energy.

June believes that organized reuse initiatives are intrusive and unnecessary: she feels that she already reuses plenty. But she will go along with such initiatives when colleagues whom she truly respects seem to believe in it.

June's guiding principle is to get the job done, whatever it takes. To accomplish this, she requires a fast workstation, a decent compiler, and a closed door. What she actually has is a not-so-fast workstation, a flaky network, and a CASE tool that she draws design diagrams with.

June's expectations from others are limited: from managers she expects hot air; from her peers she expects to hear a lot of war stories that tend to bore her.

Ross

Ross, age 50, is the senior manager in this scenario. His main responsibility, as he sees it, is to ensure annual profitability in his Line of Business. To accomplish this he needs timely and accurate information, good advice from the Director of Technology, and good employee morale.

Ross places a premium on his employees' happiness, motivation, and desire to excel; he also expects sacrifice and long hours from them. He dislikes lack of "ownership" by his employees, the attitude that this is just a job, apathy, and low standards.

Ross has at his disposal some excellent staff at various levels of the organization. He knows that his budget is inadequate for all of the technology base and infrastructure needs that his people express. He knows that competition from other companies is getting stiffer, but is not sure how to address it: there is too much technology "hype" to understand (most recently, OO) and, indeed, too much management "hype" to understand (most recently BPR). His dilemma in this drama is the perennial question: where to put the money?

The Story

Act One: Stirrings

Scene One: The Opportunity. Joe and Mike are chatting about the current state of their project when Mike observes that there has got to be a better way: he sees so much duplication of effort between projects, and it bothers him. Developers are continually creating similar system architectures and coding analogous components from scratch without seeing what could be borrowed from previous or concurrent projects, making the same mistakes and learning the same lessons over and over.

Joe responds by agreeing in principle — but points out that he has seen this situation many times before, and he knows that the project managers will never buy into a plan for developing common components. Their jobs are complicated enough as it is.

Mike, ever on the lookout for ways to innovate, is not discouraged. He shows Joe a journal article about a new design-for-reuse technique and describes how he believes the technique can work in their organization.

Joe is open to trying, so he drafts a memo to Jim, their Project Manager, describing the opportunity and its potential benefits.

Jim knows about Janice from a presentation she gave about reuse to a group of managers. He forwards the memo to her with an annotation that "it's probably a non-starter, but let me know what you think."**<1>**

To Jim's surprise, Janice thinks it is a great idea. She sends e-mail to Joe and Mike (copying Jim) to propose a meeting.

Scene Two: Exploration. At the meeting, Janice hands out some of her stock reuse literature. On the basis of what she has heard from Joe and Mike about commonality between projects in their Line of Business, she conjectures that there may be an opportunity for domain engineering. She suggests that they begin to sign up others to support the idea.

Joe suggests that they try in particular to sign up Project Managers since that has been the source of resistance in the past.

Mike proposes that they talk to other software engineers, who will understand and appreciate the opportunity for reuse.

Janice's perspective is somewhat different: she has a strategic vision for reuse throughout the Division.**<2>** She thinks that Andy (the marketing person) should get involved as soon as possible to promulgate understanding of the competitive benefits of a reuse program — and that selected customers should also be brought into the huddle.

Janice's ideas make Joe nervous. He believes it is too early to bring Andy in, let alone customers.**<3>**

The three agree to talk the ideas around, and to hold a next meeting involving a larger group.

Scene Three: Broadening the Base. The three begin to hold one-on-one conversations about their ideas with prospective supporters. Joe approaches June, a colleague he has known for a long time. Since June works on a different project, Joe figures, signing her on will help spread support for the reuse program.

June is skeptical, but she signs on because she sees the potential benefit and respects Joe's opinion.

Janice reports to Jim on the initial meeting and invites him to the next meeting.

Jim agrees to come — he is surprised that the ideas are moving forward, and indicates that he sees no harm in them. However, he does not show up at the meeting.**<4>**

Janice also invites Andy, who does show up. Joe is not happy about this.**<5>**

At this second meeting, the group discusses possible experiments that could illustrate the benefits of reuse. They develop a plan to perform an experiment and, if it is successful, to use it as the basis for a pitch to management for increased support. The experiment involves adapting three existing software modules so they can be reused in multiple systems.

Act Two: Beginning Changes

Scene One: An Experiment. To get the initial experiment going, the group enlists the support of other Project Managers. With Janice's influence and the help of the Project Managers, they obtain a small amount of R&D dollars to fund an experiment. The goal of the experiment is to develop a small number of components to be shared by several projects.

Technical meetings are now held, in which representatives from the different projects participate. The participants decide that their technical approach will be parameterization, and they begin hacking. The resulting components are a modest generalization of the types of components previously developed in these projects. They appear to work, and the experiment is declared a success.

Scene Two: Sussing the Brass. The technical team are excited about their initial success and want to continue along those lines — perhaps developing additional common components.

Janice urges a grander view of a Division-wide (or at least Line of Business-wide) domain engineering program.**<6>** Janice and Andy meet with Ross, the Line of Business Manager, and separately with the Division Head. Their goal is to judge the receptivity of these managers to the idea of a larger reuse program.

Scene Three: Planning the Pitch. Janice reports back to the technical folks that the Ross is sufficiently receptive. She urges the group to prepare a pitch for a domain engineering initiative. She suggests that the principal argument in favor of the initiative be Return on Investment, emphasizing the savings in project development dollars and the consequent increase in competitiveness. They should request a management commitment to establishing a reuse-based product line initiative in the Line of Business, including a domain analysis and development of an architecture and asset base.

With all of this excitement, Joe, Mike, and June are beginning to see themselves as being used by Janice and Andy.**<7>**

Jim, in the meantime, is beginning to feel bypassed. He is nervous that the flurry of reuse-oriented activity will pull resources away from his project.**<8>**

Joe, Mike, June, Janice, and Andy all take part in preparing the pitch to the Ross. They structure the presentation in terms of:

- The opportunity
- The return on investment (ROI)
- The risks
- The recommendation

They decide that the technical team will present the opportunity. Andy and Janice will cover ROI: Janice will present the Software Engineering Institute's cost model for software reuse; Andy will speak about the ability to bid jobs more cheaply and thereby bring in more contracts. They try to anticipate Ross's objections. They identify possible risks including:

- The waste of precious R&D dollars
- Tying up key engineering staff who are desperately needed on customer projects
- Compromising on "-ilities" in order to reuse an existing component

They develop responses for each of these issues. Joe, as the most seasoned engineer in the group, is tasked with responding to the last of these risks so that the proposal is technically credible.

Scene Four: Management Signs On. As prepared as they were, the presenters are hit with some questions they did not anticipate. Ross raises some expected concerns about Janice's ROI arguments, but surprises the presenters by saying that he does not understand how what they are proposing — this notion of reuse — is any different from what software engineers in the division are already doing.**<9>**

Ross asks them why object-oriented methods do not already solve the problem, as he had been led to believe when approving last year's OO R&D investment, influenced largely by the recent onslaught of OO literature targeted to management.<10>

He does not understand the idea of domain analysis at all.<11>

Finally, Ross suggests that if the ideas are so good, the team could get an R&D contract with the Government to support the work.<12> Janice responds to this last point by recounting her experiences in trying to use contract R&D for such purposes.

Ross finds merit in the proposal — even if he does not fully understand it — partly because of his long-time respect for Joe. He does not commit to a full-fledged product line initiative as requested, but he approves funding for a pilot domain analysis project,<13> and secures a commitment by two ongoing development projects to participate in the pilot.

Act Three: New Conquests

Scene One: The Pilot. Although they put themselves fully into the presentation, Joe, Mike, and June are still feeling put upon by Janice. They feel like her "technical guinea pigs".

Janice's approach is very tool and demonstration oriented, a consequence of her years doing contract R&D. She has a reputation for pushing glitzy methods, and becomes more and more rhetorical with time. Her overall approach to technology transfer is based on push rather than pull, and this does not appeal to Joe, Mike, and June who come from the trenches.<14>

In reaction, these three start taking shortcuts with the domain analysis method.<15>

It becomes more and more apparent that there is a distinction between a true pilot — an effort intended to enlighten the participants and lead the way towards change — and a "sandbox" project which is insulated from real world concerns. Certain factors pull this effort towards the sandbox category: there is no clear linkage to the final customers of the software; as a result, the Line of Business Project Managers begin to resist because they do not see it as being in their interests. They see their personnel resources diluted for the sake of the pilot project and the dog-and-pony shows that Andy arranges. In reaction, the Project Managers begin to retract their resources, calling technical personnel back to assist in handling project emergencies.

The pilot team reaches out to establish a link to one particular customer, and this liaison becomes the basis for most of the technical decisions. The other participating project bails out, justifying their action in terms of schedule: the results of the domain analysis were not ready in time for them to be used.<16>

Despite these difficulties, the pilot effort does produce a number of code assets, which are used by the remaining participating project. That project gives the pilot effort high marks for having saved it significant development cost. This assessment is met with some skepticism elsewhere in the Line of Business, however, because it did not consider the cost of the pilot itself. Furthermore, the use of components developed through internal funding raised a host of ownership questions that the LoB's contracts people now have to negotiate with the project's customer.

Scene Two: Scaleup Difficulties. The success of the pilot domain analysis, such as it was, turned out not to be transferable on a larger scale. Other Project Managers resisted adopting the asset base, for several reasons:

- They did not believe that the assets would meet their projects' needs, since these were not

considered during the domain analysis<17>

- They were concerned that they would not have influence over the maintenance of the assets<17>
- There was no clear statement of the context in which the assets were meant to be used (a consequence of methodological shortcuts taken during the pilot domain analysis).

Some of the projects tried to use a couple of the assets, but because the assumed context was not defined, the attempted uses were inappropriate and led to errors. This experience reduced the project managers' confidence in the assets.<17>

In the face of Project Manager resistance, the Line of Business Manager did not go to bat for the asset base,<18> and the initial technology success failed to scale up.

Scene Three: Enter the Gladiators. With the lack of demonstrable scale up, the reputation of software reuse was waning within the Line of Business. The opposite, however, was true within the company at large.

Urged on by recent Government initiatives and mandates, and a slew of executive summaries about the benefits of reuse, the company's top management — just around this time — instructed the corporate software technology group to institute software reuse around the company.<19>

As we leave the scene, representatives from this group are meeting with various members of our Line of Business, "introducing" them to the ideas of domain analysis and taking notes about the experiences we have just described.

4.2 Scenario Analysis

In the scenario, several key events were identified. These events are analyzed and interpreted in the commentaries below.

Event 1: Non-Starter

Jim's offhand comment — that the proposal from Joe and Mike is probably a non-starter — raises questions about how reuse opportunities are identified and assessed. Frequently the motivation for such remarks is difficult to determine: Is it a knee-jerk defense against anything new? An implicit lack of trust in the judgment of Joe and Mike? An expression of pessimism about software process change in the company (or in general)? A belief that if the idea were any good they would already be doing it, or he would at least be hearing about it from his peers or management? An underlying belief that a lot of things that "would be good in principle" can in fact be harmful if they interfere with the established working routines, schedules, and resource allocations?

As technology transfer "in reverse," reuse opportunities can be easy to miss. They can begin anywhere in a software development organization: they may grow from the bottom up, rather than as mandated initiatives or large-scale investments. Like the shoot of a plant, however, such bottom-up growth can be thwarted easily through neglect.

Belief maps can help in understanding what motivates a manager to sign on to, condone, tolerate, neglect, or discourage a reuse opportunity. Continuing the horticultural metaphor, does the manager view the "plant" as a source of nutrition for the organization's work, as a mere decoration, or as a weed? If he sees it as a decoration — nice but not essential; not of direct, quantifiable value — will he tolerate it because it might improve morale? At what cost? And is "toleration" an

effective death sentence? Or does the manager see the proposal as neither nutrition, decoration, nor weed, but as a combination of features that has no chance of surviving — an evolutionarily unfit species?

Imagine that if in the scenario Jim had been gung-ho on the opportunity and Janice were cool. Then what would we wonder about Jim's motivation and underlying belief map? Does Jim see this as an unusual opportunity to satisfy his customers? Is he bending over backwards on this occasion to keep Joe happy? Or maybe he doesn't like Janice and wants to send her on a fishing expedition.

Whatever Jim's motivations and beliefs are, his behavior regarding the opportunity permits rich interpretation. Grounding such interpretation by reflecting on underlying beliefs can be especially important in teasing out patterns early on which may later appear in a more fully developed form. Articulating beliefs and their reasons, then evaluating how well those reasons match the facts, can go a long way towards effective analysis of reuse opportunities.

Event 2: The Grand Vision

For someone who has bought into the ideas of software reuse — especially one who has recognized the non-technical aspects of the problem — it is tempting to see reuse as a large-scale phenomenon, necessarily involving cultural change across the organization. The real challenges, after all, concern scale-up. Competent software developers have always practiced reuse in a private mode: they build their own function libraries; they adopt and refine their own processes to accomplish certain types of tasks. On occasion, they will share what they have with colleagues who seem receptive. The real problem is institutionalizing such practices. So, someone like Janice will argue, does not "systematic reuse" necessarily imply "organization-wide?"

The flip side of this issue is that the software engineers, who are already practicing private reuse (or even reuse on a project-wide scale), may see such grander notions as just so much hot air. They might believe that they are already doing what is necessary; or, as in the case of Joe and Mike, they may see an opportunity for improvement but their vision will still be firmly rooted in the day-to-day realities of their assignments. They know that although private reuse already occurs, it is not trivial and requires substantial thought, experimentation, and skill. They know that the technical issues which arise are too specialized even to be expressed, let alone explained, to management. They will, therefore, view designs for organization-wide change with suspicion and skepticism.

There is justification for both of these viewpoints. Underlying the grander view is the belief — justified by much empirical evidence in the industry — that private reuse does not significantly impact the crisis of software engineering. Underlying the view from the trenches is the belief — justified by hard earned experience — that reuse is difficult, that there is an art to creating reusable software, and that the process cannot scale up simply with a wave of the hand or a management endorsement of organization-wide reuse.

The fact that there is truth in both positions can lead to stalemates that frustrate reuse efforts at all levels. As shown in the system diagram in Exhibit 5, each side has an argument that — as long as it stays an argument — effectively defuses the other side's program. This defusing can be immediate and overt, resulting in aborted initiatives. Or, as in our scenario, it can be drawn out over the life of a reuse program and (intentionally or not) reduce the program's effectiveness.

Alternatively, the tension in this conflict can be converted to fuel for learning that actually drives the reuse program. The Knowledge Evolution Grid introduced in Section 2, which identifies two dimensions of progress towards a knowledge creating organization (private-public, implicit-

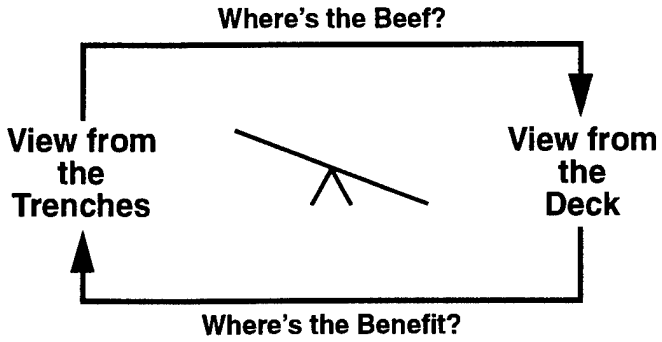


Exhibit 5. View from the deck vs. view from the trenches

explicit), provides a useful tool for accomplishing this conversion. The two positions, in the trenches and on the deck, are in fact advocating progress along different axes. The engineer in the trenches sees clearly (although he is unlikely to state it this way) the challenges in moving from tacit to explicit knowledge. For example: defining an interface that satisfies successive application contexts involves articulating the commonality in the applications; this commonality may have been intuitively apparent for some time, but articulating it is not easy. Engineers like Joe and Mike recognize the difficulties, but also appreciate the benefits; hence they advocate focusing on progress along this path. The visionary on the deck, like Janice, understands both the challenge and the need for communication as part of reuse. Hence she directs her gaze along the private-public axis.

The winning insight is that progress is needed along both axes. This observation can turn the unproductive conflict of our scenario into creative tension.

Reflection on Resistance: "Reuse is unnecessary overhead"

The discrepancy between the view from the trenches and the view from the deck accounts for much of the resistance to reuse as a thing imposed from above. Consider, for example, the following scenario: a reuse advocate (who in our story could be Janice, Joe, or Mike) is urging other developers to package their output for dissemination and reuse. One frequent response is, "It is not necessary." Why is it not necessary? Perhaps because:

- *"We're productive as is," or*
- *"We already practice reuse without the hype and trappings," or*
- *"There aren't that many opportunities to reuse."*

Behind all these responses is a belief that what is being proposed (or imposed) is an "overhead" activity, which will not directly contribute to software development. Proceeding with a reuse initiative without addressing these beliefs runs the risk that developers will lose their sense of task ownership, and the overall alignment of the development team will break down.

Event 3: Bringing in Andy

Who does one involve in reuse planning, and how soon? At some point all stakeholders must be involved, at some level. But in the early stages there is a tradeoff between the risks of exclusion (not involving those who hold an important stake in the outcome, or who can positively influence the process) and the risks of inclusion (involving too many varied points of view, goals, and assumptions, which may prevent the group from converging on a plan of action).

Janice sees institutionalized reuse as a major shift in the company's operating mode. She expects that the entire business model of the company might be affected, and wants such issues to be considered explicitly from the beginning.

Janice's invitation of Andy, the marketeer, reflects her knowledge of the company's business reality: management will not invest in reuse if there is no clear return on the investment (ROI). The ROI can only be assessed by considering the impact of reuse on the company's customers. Since marketing is the liaison between developers and customers, Andy's presence is needed. It might even be possible to obtain direct funding from a customer to support the initiative; again, Andy is the person to assess this.

Mike and Joe may see Andy's presence as a potential diversion from the hard technical issues. They might fear that too much emphasis on the business questions might result in an overly simplistic plan: it may be sufficiently attractive that the non-technical stakeholders will shake hands over it and then expect that the problem has been solved, thereby foisting unrealistic goals on the software developers.

Each participant is, at this point in the scenario, being driven by beliefs founded on their view of the harsh realities. The problem is that, like the seven blind men and the elephant, each participant has a restricted view of reality. In such situations, the beliefs themselves cannot be argued out to any real resolution. The facts underlying those beliefs must be articulated, and then used to construct a new model that encompasses both the technical and business perspectives.

Event 4: Manager's No-Show

The immediate question raised by Jim's absence from the meeting is: Why? We have already speculated on the reasons for his "non-starter" comment; his "no-show" may simply be a reiteration of this. But it may not be. There is an entire spectrum of possible explanations. Perhaps he really is resisting the initiative ... passively, by not showing up, and maybe he does not even realize that this constitutes resistance. Perhaps he supports the initiative but his commitment is only half-hearted, and when other events require his attention he gives them priority. Perhaps, however, he strongly supports the effort since hearing Janice's response, but other matters that really do take priority have prevented him from attending the meeting.

These are unknowns to the newly forming reuse team. Jim's absence indicates the possibility of a problem down the road. If that indication is accurate, understanding his guiding beliefs will become important for the success of the effort.

Asking Jim why he did not attend may provide answers, or it may not. Engaging him directly in an inquiry, in private, to draw out his reasons for not coming and not responding will be difficult, but may be crucial to obtaining his future support of the project. Joe may be the person most well-positioned to do this, but his skills at obtaining a non-defensive response from Jim are questionable. Generally, resistance is the indirect expression of some harsh reality. Can Joe elicit enough information from Jim to reveal the deeper meaning?

One approach using the Ladder of Inquiry is for Joe to articulate his own reasoning to Jim for why this project has potential for him and also perhaps to reveal his own doubts. This will invite Jim to acknowledge his behavior and perhaps disclose some of the reasoning or "data" behind it.

Making a move of the type described above can be quite important at this point in the project because often the cycle of broken promises becomes very quickly a vicious one that continually lowers everyone's expectations of each other. The more everyone accepts this lowering, the more

difficult it becomes to see that this is what is happening. Hence, the more difficult it becomes for Joe (or anyone) to inquire about Jim's nonattendance.

The system diagram in Exhibit 6 shows two self-reinforcing patterns of requests, promises, and follow-through on commitments:

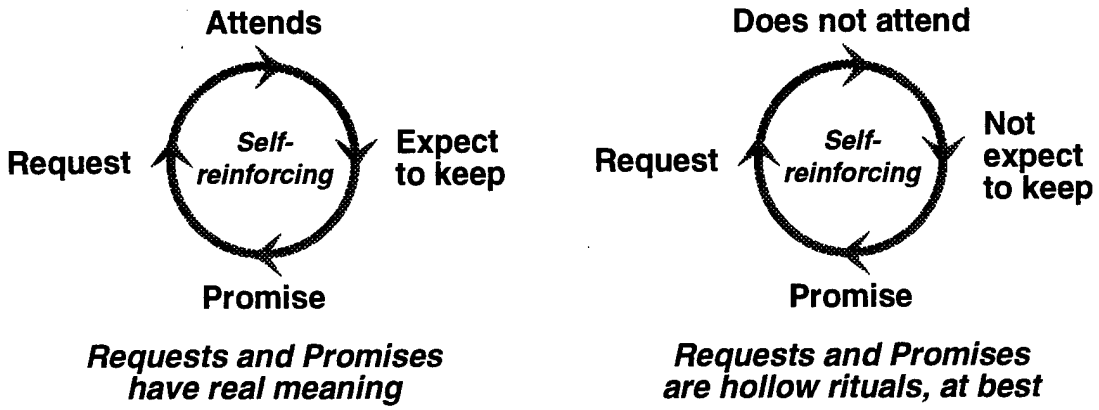


Exhibit 6. Self-reinforcing request-promise patterns

Especially at this formative point in a team's development, addressing failed commitments like the manager's no-show can set an important tone for how all conflicts are handled on the project. This is a critical moment in determining: (1) how conflicts will be handled, and (2) how to obtain support and what kind of support is actually needed from management.

Event 5: The Undesired Guest

We have already discussed the difference in perspective between Janice on the one hand, and Joe and Mike on the other. As Janice invites Andy to the meeting and Andy accepts, we see the differences escalate from perspective to action. Besides the difference in opinion, we now see a difference in style: a willingness to take action behind the backs of one's colleagues. Since the goal of the initiative is to institutionalize reuse — a form of knowledge sharing — Janice's withholding of information at this point is particularly ironic, a possible indication of "not walking the walk."

The breaking of trust at this point in the story will be a factor in the mixed success that follows. It sets in motion a dynamic, depicted in the system diagram in Exhibit 7, in which individual agendas work counter to the espoused goals of the team.

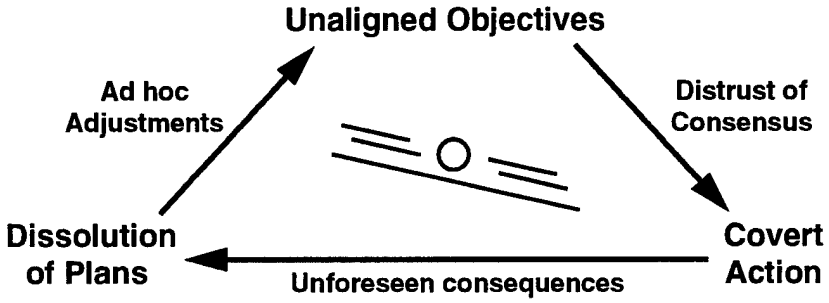


Exhibit 7. Vicious cycle stemming from unaligned objectives

The pattern starts off with a lack of alignment in the objectives of the team members. Some members may then fear that the team will not approve of certain actions they may wish to take. If they nonetheless desire to remain on the team, they will take such actions covertly (Janice's invitation to Andy). This leads to events and consequences that were not foreseen in the team's plans (Andy's attendance), and as a result the plans previously developed by the team (Joe and Mike's memo) become less and less relevant or applicable. If these developments are not acknowledged, the diminishing relevance of the espoused plans will be compensated by ad hoc measures, which can undermine the assumptions on which the team's very formation was based.

The self-reinforcing aspect of this pattern comes from continuing silence about the members' divergent goals. Since the diversity is not expressed, there is no opportunity to manage it through, for example, continual realignment.

In our scenario, part of the problem is that the goals of the team have not yet been clearly articulated. The effort is, after all, still in an initial exploratory stage. Our analysis shows, then, how important it is to articulate stakeholder goals early in the planning process, and to reassess them continually.

Reflection on Resistance: "Reuse requires too many resources"

The connection of this event with reuse as knowledge sharing is worth taking seriously. Argyris writes of the way issues and positions become "undiscussable" in an organization because the participants wish above all to avoid embarrassment — their own as well as others. When this dynamic occurs in software development, it reveals patterns that work counter to reuse.

Let us again consider the scenario in which a reuse advocate (e.g., Janice, Joe, or Mike) urges his/her colleagues to package their products for reuse. In addition to the response that "It is not necessary" (discussed in an earlier interlude), another common response is "There are not enough resources." Such responses usually contain an element of truth, but also contain hidden meanings, which make them resistant to dialog.

Diverting scarce resources can increase development pressure, leading to products that are less, rather than more, reusable. The developer is, in effect, saying "If I have to spend time packaging my product for other developers' consumption, I cannot spend that time designing it for maximal value to the customer." This argument pits the interest of other developers against that of the customer — probably a shortsighted view since the customer will benefit from cost-effective maintenance of the software. It is a skillful response, though: by shifting attention to a supposed conflict between two other parties, the developer removes himself from the spotlight.

This pattern serves to conceal two types of information: 1) tradeoffs between immediate customer satisfaction and long-term reusability, and 2) design decisions intended to maximize customer satisfaction within the constraints of a project's schedule, budget, staffing profile, development environment, etc. The truth in the developer's response is that resources are limited and, therefore, compromises are probably necessary. The falsehood in his response is that numerous policy decisions are not subjected to resource constraint arguments. They are viewed as necessary priorities. The allocation of different tasks is a value-based decision, disguised as a decision driven entirely by resource constraints.

The resistance, seen in Exhibit 8, is air-tight: with every privately made decision, the framework of design commitments and constraints becomes more restrictive, narrowing the range of choices for future issues. This in turn raises the stakes when a decision turns out to be wrong: although there is a need to reevaluate the decision, the cost of reversing all dependent decisions has risen dramatically, and so has the potential for embarrassment. All participants, then, have a stake in maintaining the "undiscussables."

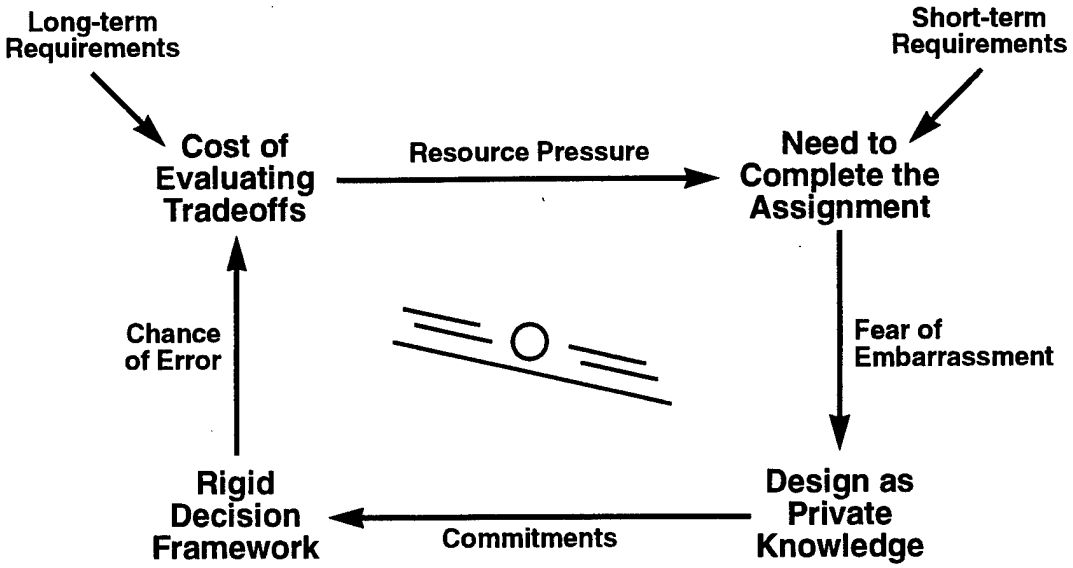


Exhibit 8. Resource pressures reinforce resistance

Event 6: Janice Urges a Big-Picture Approach

Janice's urging of the big picture is an escalation of the issues discussed in the commentaries on events (2) and (3). It is interesting to see that despite the success of the initial reuse pilot project, the positions of the team members have not converged. In fact, the small success can have the opposite effect, reinforcing the divergent beliefs of each team member. With their underlying beliefs now confirmed, each member views the next logical step according to their original agenda. Success, then, can widen a rift rather than close it.

The system diagram in Exhibit 9 describes this dynamic and shows how it interacts with the pattern discussed in the commentary on event (5). The reinforcement of beliefs that success brings causes each member to believe more strongly in his or her original agenda. As long as they keep their differences visible and make a conscious attempt to find common ground, there is potential for coordinated action. But if the differences become too wide or positions become held too strongly for compromise, the unproductive pattern discussed in commentary (5) can take over.

The least stable state of the unproductive cycle is when plans start to dissolve. That instability provides an opportunity for systematic (rather than ad hoc) adjustment, which if taken can lead back into the productive cycle.

Finally, the potential for instability in the productive cycle can be reduced if private learning is replaced by team learning. This is shown in Exhibit 10: when the perspectives of all members of the team are applied together to draw lessons from a successful project, beliefs may be modified instead of reinforced. By collectively drawing conclusions from a common experience, the team can become more cohesive and the agendas more, rather than less, aligned.

Event 7: Joe and Jane Feel Used

Joe and Jane's response is another symptom of the differences in vision discussed above. The fact that the team continues to move forward reflects their recognition that, despite the different agendas, the two "camps" depend on each other. Joe and Jane need Janice to represent them to

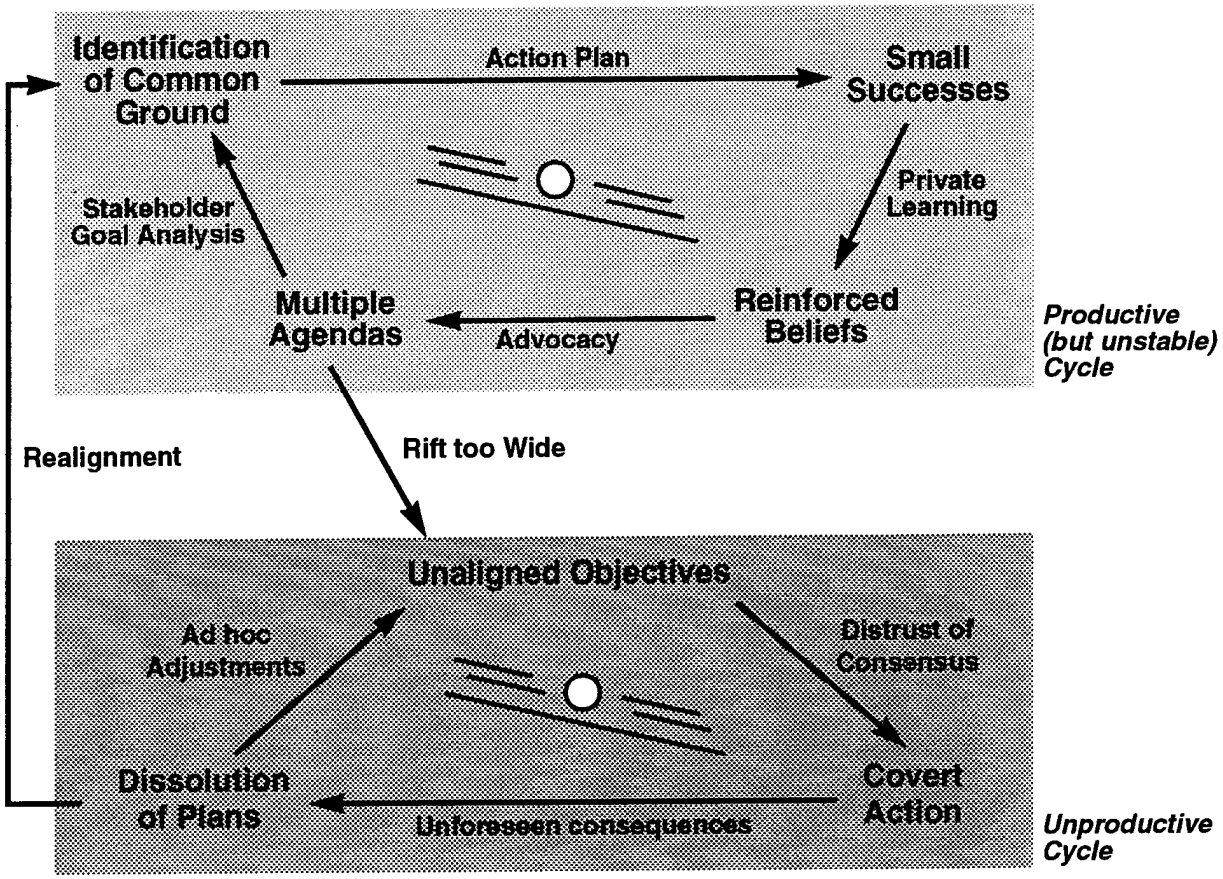


Exhibit 9. Small successes reinforce divergent beliefs

management — or they think they do. At the same time they wonder whether she is a "loose cannon" given to over-ambitious declarations which they, as the engineers, will have to make good on. They are worried that her real goals in this initiative have more to do with empire building than improving the company's business processes. If so, are they simply her pawns? Or are they overestimating Janice's power, and her hunger for power? Janice, in turn, needs Joe and Jane to give her credibility among the organization's engineers. We do not see any sign, however, that Joe and Jane will use this leverage to temper Janice's vision with their own.

We see in how the undiscussables become food for speculation about motives. The speculation reinforces feelings of disempowerment, which in turn fuel unspoken resentments, which raise the stakes and make the issues that much more difficult to discuss.

Event 8: Joe's PM Feels Bypassed

The ambiguities discussed above in relation to Jim's no-show have now escalated to the point at which they can adversely impact the reuse initiative. As the initiative expands, the buy-in of project managers will become more and more crucial. But Jim has never explicitly bought in, nor has he explicitly rejected the initiative.

All we can say about Jim's position in relation to the initiative is that it remains ambiguous, both to us, as outside observers, and to the reuse team ... perhaps even to Jim himself. His concerns about meeting project milestones and having resources taken away from his project are fully

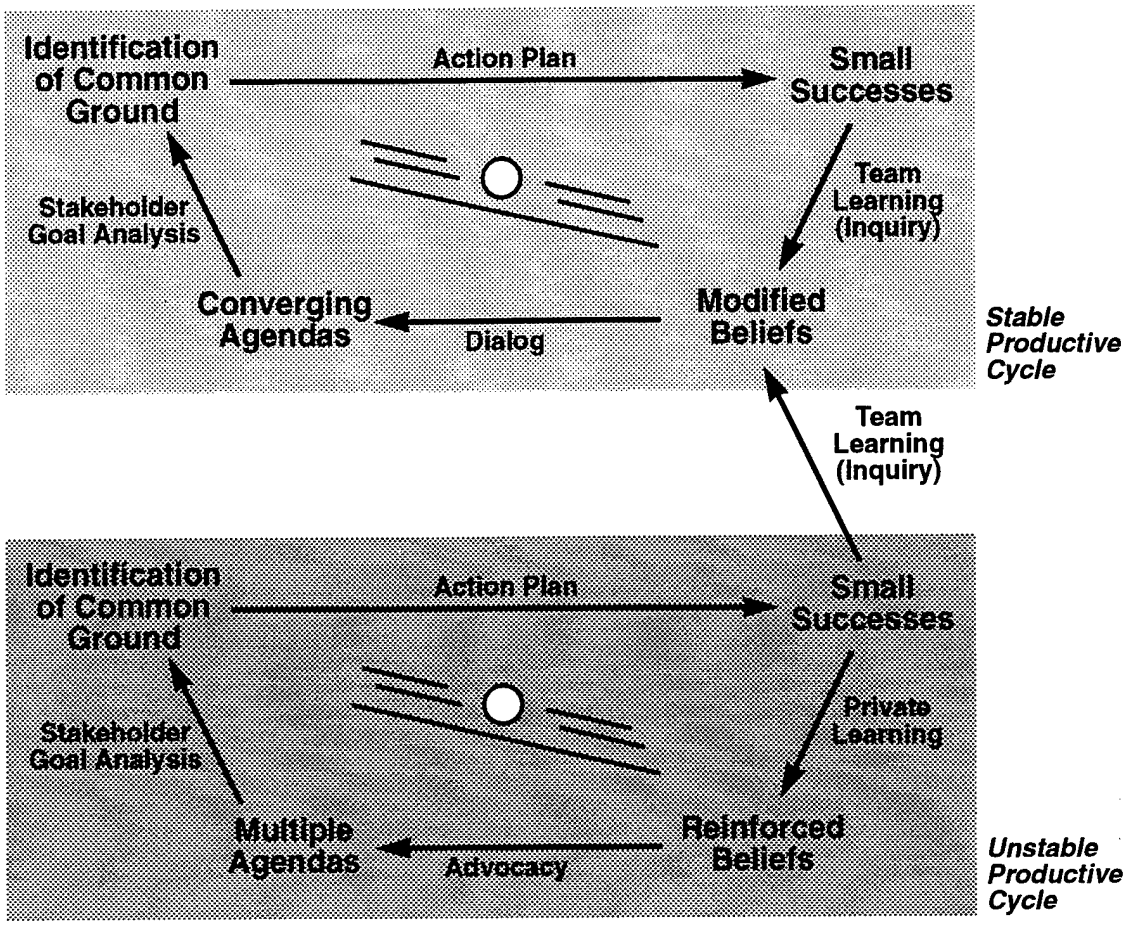


Exhibit 10. Team learning replaces private learning

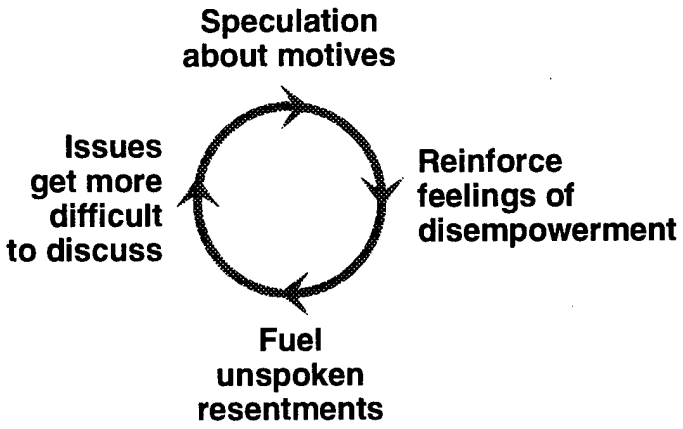


Exhibit 11. Undiscussables fuel a vicious cycle

understandable. He witnessed such problems even during the initial experiment. But is it also the case that he simply feels left out of the process — that he is not viewed as a stakeholder in the reuse initiative? Janice's method of keeping Jim nominally in the loop by sending him periodic email may be backfiring, giving Jim the sense that Joe and Mike are now reporting as much to Janice as to him.

The reuse team is probably speculating a great deal about the reasons behind Jim's no-show. Such disconnects can escalate quickly and point out that all was not well before. This is clearly a place when Janice or Joe would help the project a great deal by using the Ladder of Inquiry to directly find out about Jim's reasoning for his behavior. Given Jim's focus on his projects and his impatience with anything else, it is the team's responsibility to check more frequently and more thoroughly with him about where he stands vis-a-vis this project.

This is a case where a laissez-faire attitude may be adequate, and active support may not be required. However, it is also a case in which a reuse team needs to have been rigorous about stakeholder analysis in order to identify who can hurt the effort if a stakeholder's needs (stated or unstated) go unmet. Luckily, Jim is just getting nervous and perhaps flexing his muscles so far, but he has not pulled Joe or Mike off the project ... yet.

Event 9: Nothing but Good Software Engineering

A lot of attention is being paid these days to quantifying the return on investment in software reuse [Gaff89. Crui91, Barn91, Rief91]. The reason is clear: investment in reuse competes against investment in direct value producing efforts. If reuse is seen as a means of process improvement rather than as a direct value producing endeavor, the burden of proving ROI will lie heavily on the advocates of reuse. The competition between the two forms of investment is summarized in the system diagram in Exhibit 12:

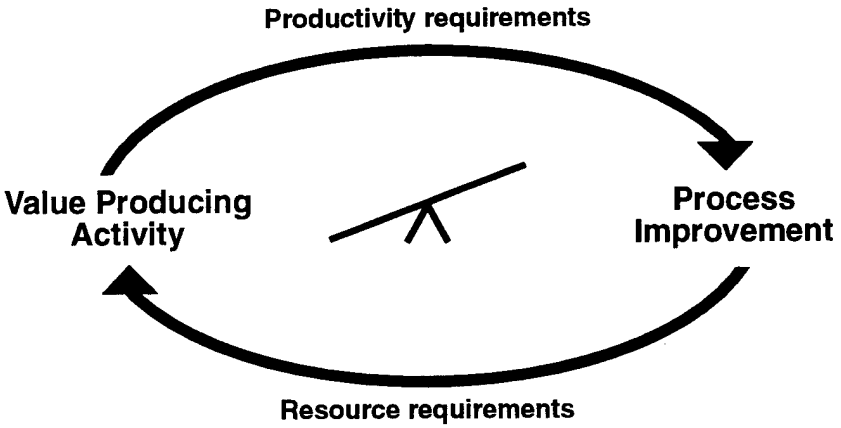


Exhibit 12. Competition between application and infrastructure investment

Janice was right in expecting management to challenge the team with this issue, but she did not foresee all of the related arguments. ROI arguments are vulnerable for several reasons. First, there is not a lot of data to support them, because there has not been long-term comparative measurement of processes with and without reuse. This is partly because reuse as a recognized discipline is still fairly new, and also because researchers are still trying to decide what the appropriate measurements should be.

Second, ROI arguments are always speculative, in any endeavor. The listener knows full well that the best "spin" and most optimistic interpretation are being put on the available data. This knowledge, however, remains undiscussable since the ostensible purpose of the ROI presentation is to provide solid, credible predictions. Thus, if the listener is not predisposed to invest in the venture, ROI arguments are unlikely to convince him.

Janice also did not foresee the argument that reuse is "nothing but" good software engineering. Coming from management, this argument is remarkably similar to the software engineer's response discussed earlier: "we already do this." Both responses demonstrate the trap of regarding reuse as a technology that must be promoted and transferred, rather than technology transfer in reverse (as discussed in Section 2). The more one argues the issue in terms of conventional technology investment and transfer, the weaker one's position becomes, because the thing one is promoting is so intangible. To quote Gertrude Stein, a noted literary change agent, "there is no 'there' there."

When a presenter is caught "flat-footed," it is a sign that the belief systems of the listeners have not been adequately considered in developing the presentation. In our scenario, careful thought about the LoB manager's belief system might have revealed the need to present reuse as a measure that directly creates value (as opposed to indirectly, by improving processes); and to present it as technology transfer in reverse rather than something new to be foisted on software developers.

Event 10: Didn't OO Already Solve Our Reuse Problems?

Managers who have been sold on object-oriented (OO) technology are often confused by the promotion of reuse as something distinct from OO. Less technically savvy managers may have bought into OO as the "silver bullet" that was supposed to transform software development from a craft into an engineering discipline [Cox90]. To such a manager, the appearance of reuse as "the next latest thing" may seem like technological bait and switch: "We know we told you that OO was going to solve your problems, but now we're here to tell you that what you really need is reuse."

Managers who understand something about OO may have assumed that objects (as opposed to other types of software components) are intrinsically reusable since they have well-defined interfaces. This position has been put forward by technical advocates of OO, and it should not be regarded as an expression of technical ignorance. Arguing to the contrary — explaining why even a well-encapsulated object is not necessarily reusable — requires subtle software design considerations and is not easily communicated to management [Berl90]. Arguing that even reusable objects may not actually be reused involves broader considerations such as those discussed in [Gris95].

In the OO community itself there is skepticism about reuse as something different from, or value added to, OO [John95]. Belief mapping and the Ladder of Inquiry can be effective tools in defusing this clash of communities and self-interest by shifting the focus onto the technical and empirical facts: how and when software is reused. Conveying the point to management can be more difficult, though, because the detailed technical facts cannot be brought to bear.

This is one place where the grander view held by Janice and her ilk can be effective in winning management over. The idea that reuse is more than constructing reusable components, the notion of reuse as knowledge creation and sharing, the view that reuse is technology transfer in reverse — these concepts are potentially meaningful to someone who thinks in terms of business models. The challenge is to formulate the argument in those terms, starting from the models with which the manager understands his or her job reality.

Event 11: Clueless in Babylon

Pitching domain analysis to a management audience is a challenge. If the problem in selling reuse as a concept is that there is no "there" there (other than OO, which "we already do"), then consider how lost a management audience will become when the talk turns to modeling commonality and variability in a domain. Is it reasonable to expect management to understand these concepts? A manager may never have suffered through trying to bend a rigid software interface. He may never have been forced to reinvent the wheel because the last one built was the wrong size. Can someone, even if quite intelligent, really get the significance of domain modeling and clearly see the distinction between reuse practice and OO technology if they have never experienced the deflation of learning that the new whiz-bang component they just built is totally at odds with the design assumptions of a fellow developer?

The problem is not that the listener is unintelligent. It is that his experience is different from the software developer's. The reuse advocacy team, as the petitioner in this process, cannot expect management to come out and meet them on the turf of technobabble. Failure to meet management on their conceptual terrain may be enough, in some managers' eyes, to disqualify the presentation from the start.

The scenario technique that we are using here, in this document, is one way to bridge this gap. The concrete role-to-role interactions and decisions are common experiences that everyone in the organization can relate to. By keeping the scenarios fictional, the emotional temperature can be kept down, at the same time allowing each participant to see herself in the story.

Event 12: Why not CR&D?

The suggestion that the reuse initiative could be funded through a contract should serve as a warning sign to the presenters. It could well indicate half-hearted commitment on the part of management: "We will support the effort as long as it does not cost us anything." This can be a form of resistance similar to the proof-of-ROI cycle discussed above. At worst (as happens in our scenario) it can lead to efforts undertaken without sufficient resources and without a commitment by management to follow through in the long term. This is a setup for failure.

Alternatively, the CR&D suggestion might indicate a sincere desire to minimize the cost of the initiative, without implying a lack of real support. But such a position reveals a misunderstanding of the initiative. Contract R&D could certainly support the development of method and tool prototypes, and even their pilot application. But technology development and evaluation is not the goal of this initiative (although it may occur along the way). The goal, rather, is to change the way business is performed. Contract R&D by its very nature cannot effect such change: its course is, in the final analysis, driven by the contract rather than by the needs of the performing organization.

Because the suggestion is, on the surface, so innocuous — who could argue with an approach that allows the initiative to happen and at the same time reduces cost? — it can be difficult to confront. To argue against it can be construed as weakness in the proposal, a lack of confidence that the initiative could be sold to an outside customer. Responding to the suggestion requires that it first be interpreted correctly, either as stalling tactic or as misunderstanding. In the former case, Contract R&D is not the issue at all; the presenting team must try to find the real reason for resistance, and then decide whether they can address it. In the case of misunderstanding, it becomes important to stress, again, the idea of changing the business model through technology transfer in reverse.

Reflection on Resistance: Surfacing the Hidden Reality

The team has tried to anticipate concerns and objections in their planning. In the actual presentation, however, they encountered problems for which they had not planned. These came from a few different areas. In dealing with objections and concerns, it becomes a challenge to distinguish "good faith" concerns from resistance. At first, it is most appropriate to take the issues at face value and seek clarification, address them with more/better evidence, or communicate additional information at a future time. However, when the same response persists repeatably despite reasonable attempts to directly address the concerns, the responses may be construed as resistance.

Some resistance can be quite overt, stemming from basic differences of opinion about the effectiveness of an approach. More typically, resistance is covert, involving some objection or concern which a person does not want to state directly; this form of resistance is quite difficult to identify and deal with. Such resistance is an indirect expression of a harsh reality. Harsh reality is a strongly motivating problem that influences thinking and behavior and does not easily go away. Articulating a harsh reality takes skill and trust; most people don't advertise their harsh realities readily. Rather, what usually happens is that the harsh reality is expressed in some indirect form. This keeps it private and makes the behavior which is visible quite difficult to interpret and respond to appropriately.

Covert resistance is manifested in many forms, such as active objection, changing the subject, attacking the method, silence, or even half-hearted agreement. The team in our scenario cannot yet tell if it is meeting resistance to reuse in general, or resistance to any activity that is not directly revenue-producing in the short term, or resistance to anything new. They do not know ... yet.

Using belief mapping and the Ladder of Inquiry, they might have been able to close the gap between the expressed objections and the real concerns of the management. What has happened is that the real concerns (harsh realities) may be undiscussable, so that the objections and the responses take place only in the realm of what is discussable — the stated objections. This may be even more true when the managers are with each other in a group than in smaller meetings with the team members. In our scenario, the stated concerns take a few forms: asserting that we already do this (there is no problem), the technology argument (but OO will handle all of this), just plain confusion, and the CR&D suggestion that "we want it if we can get it for nothing." As stated above, these are not automatically resistance unless they persist.

The team could have used these two tools to: (1) figure out if these responses were resistance or not and (2) if they were resistance, to attempt to draw out more direct expressions of their harsh realities.

Event 13: The Pilot

The undiscussed issues and ambiguous motives that we read between the lines of the presentation now converge in the outcome, the decision to fund a pilot domain analysis. The allocation of funds for a pilot can be viewed as an attempt to test the waters before making a major commitment (a form of learning), or as a setup for failure (a way of maintaining the status quo). The latter interpretation is supported by the observation that management made no longer term commitment, even contingent on the pilot's success. Specific success criteria were not identified as part of the funding decision; this suggests that the pilot was not viewed by management as a step towards a strategic goal.

The small level of funding may also indicate a misconception about such efforts, namely that they are "linear and continuous," in the sense that if you provide a fraction of the requested funding

you will get that fraction of the results. In reality there is a quantum relationship between investment and results: a certain critical amount of effort must occur for there to be any useful results at all. Making this point can be dangerous for the reuse team, as it can be construed as a sign of weakness in the proposal. Turning down a reduced level of funding can be viewed as a sign of business immaturity or rigid idealism, or as a defensive measure to avoid having to put one's ideas to the test.

All of this is the stuff of belief models, and in our scenario most of it remains unarticulated — a fact that does not bode well for the continued success of the effort.

Event 14: Janice Goes for the Glitz

Janice's increasingly rhetorical style shows the pros and cons of missionary zeal. Without such enthusiasm and commitment, it is unlikely that the initiative can overcome the organization's natural resistance to change. But when converting people to the cause becomes a driving force in its own right, the original purpose of the effort can easily be left behind. The reuse initiative then becomes a technology initiative, subject to the conventions of technology transfer and, in this case, to the notion of "technology push" rather than pull. Priority is then given to demonstrations and other marketing devices; these give investors the impression of progress, but they can divert attention from the real work to be done. The system diagram in Exhibit 13 shows how this process becomes self-reinforcing:

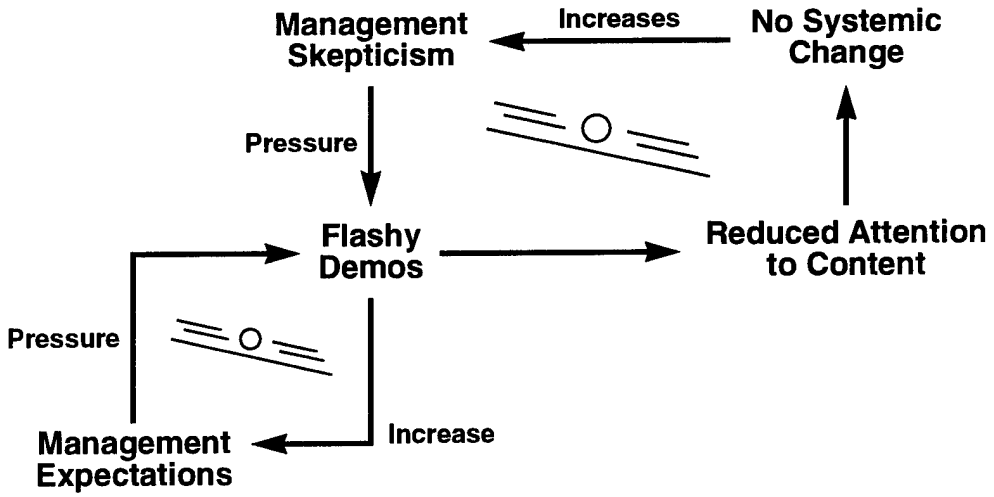


Exhibit 13. The self-reinforcing "demo-ware" cycle

Flashy demonstrations tend to confirm and increase management expectations, thus adding to the pressure to keep them happy with more demos. At the same time, the lack of attention being paid to the difficult but unglamorous issues prevents the initiative from having any substantive effect on the organization. This tends to aggravate management skepticism, which also creates more pressure for flashy demos.

Event 15: Joe, Mike, and June Take Shortcuts with the DA

The belief that "reuse is just good software engineering" has a more specialized counterpart, the belief that "domain analysis is just identifying what is common to a set of systems." All you have to do is gather some people who know the systems well, and have them articulate what's common and, by implication, what is variable.

Those who have tried it know that successful domain modeling is difficult, because of such questions as:

- Which systems do we examine?
- Whose vocabulary do we use?
- How do we represent common functions?
- How do we represent variability?
- How do we avoid bias towards particular systems?
- How much detail do we provide?
- Who do we envision using the results?
- How do we envision the results being used?

From the point of view of the engineer in the trenches, such questions can have the ring of methodology-speak. They would say that so-called technologists who don't earn their living building systems raise such issues to justify their employment. They create problems where there aren't any.

We therefore find Joe, Mike, and June taking shortcuts around these questions. This is not necessarily a sign of ignorance on their part. We have encountered this type of impatience within the reuse community itself, including some of its most prominent members.

The problem is that the motivation for considering such questions has not been made clear. In our scenario, since Janice introduced the others to the domain analysis method, it was incumbent on her to motivate its steps. Otherwise, given the limited resources of a small pilot project, and the pressure to deliver tangible results to a doubting management, any steps that seem superfluous will tend to be bypassed.

It is that much easier for Joe, Mike, and June to take shortcuts because their vision has still not been reconciled with Janice's. Faced with an apparently superfluous step, they can dismiss it as a symptom of Janice's grand vision to which they do not subscribe. Modifying the method can be a way for the engineers to reassert some authority, to regain a feeling of influence over the process, which up to this point has been dominated by Janice's agenda.

These shortcuts may succeed at making the technical team feel better, but ultimately compromise the technical work. What the team members still have not done is openly state and address the conflict. As pointed out previously, they could use the Ladder of Inquiry and belief mapping tools to do this. At this point in the scenario, we see how their continued failure to address these feelings in a direct, social interaction will begin to negatively influence the technical content of the work itself. These self-protective actions will make the real issues (i. e., differing goals and agendas) even more difficult to discuss in the future.

Event 16: One Stakeholder Project Pulls Out

The pullout of one of the two participating projects reveals one of the weaknesses of short-term reuse pilots: a client project has an interest in the pilot only insofar as it can employ the results within the project's timeframe. When there are only two such clients, the pilot is placed under enormous pressure to deliver usable results. This, in turn, can lead to premature decisions in the design of assets intended to be reusable in many future projects.

The lesson in this is that reuse is not a short-term endeavor. If a pilot is started as a way of mitigating investment risk and as a vehicle for learning, then the short-term nature of the pilot must itself be recognized as a risk, which can be mitigated by placing the pilot in the context of a longer-term plan that builds on the lessons of the pilot. There is also a simple lesson in numbers: if you need 2 pilot projects, you had better line up 3 or 4 in case 1 or 2 drop out.

There is a deeper issue as well. While some of the problems in Act Two were a result of viewing reuse as process improvement rather than value creation, here we see the opposite. The reuse pilot is expected to produce assets of provable value and to do so in a short timeframe. The problem is an overly restricted concept of "value," which is still seen to reside in consumable products rather than in applicable knowledge. The organization has not assimilated the idea that systematic learning produces value. As a result, the reuse pilot is viewed as a form of technology evaluation, reuse being one technology alongside many others, and when it fails to deliver as promised attention can shift to the "next latest thing," as seen in Exhibit 14.

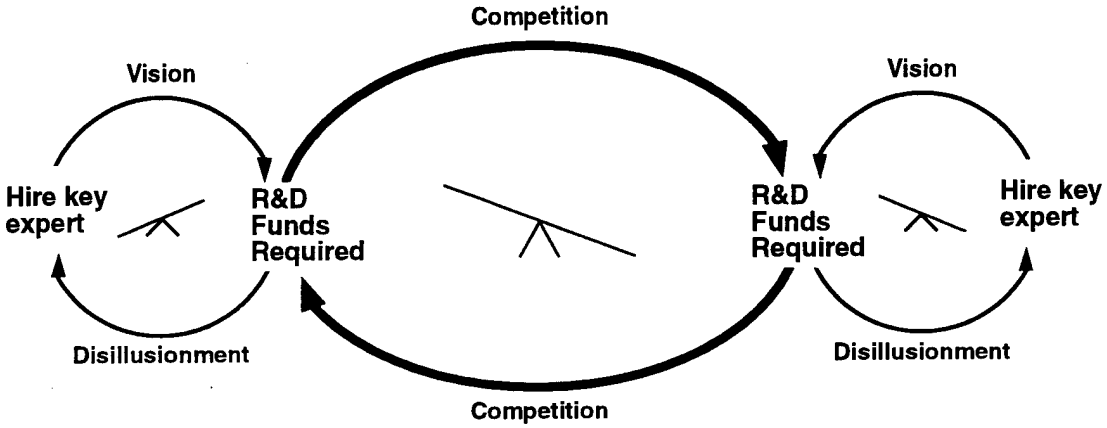


Exhibit 14. Technology investment seesaw

Event 17: Other Project Managers Decline to Adopt the Domain Model

The failure of the pilot domain analysis to scale up is an outcome of the unresolved issues we have observed: unreconciled visions, incomplete stakeholder analysis, lack of attention to domain scoping and pilot project goals. The team failed to define a business model that would address the concerns of potential client projects, specifically ownership of the reusable assets, responsibility for maintenance, and warranty of the assets' performance within well-defined contexts.

The importance of these issues is most easily seen in retrospect when attempts to scale up fail. Even among proponents of domain analysis, little attention tends to be given to the early phases in which stakeholder goals and alternative domain boundaries are considered. If we refer again to the Knowledge Evolution Grid, we can see that the focus of effort in domain analysis is still on the progression from implicit to explicit knowledge. The progression from private to shared knowledge — in this case, from relatively private (small pilot team) to more widely shared (multiple projects) — is assumed to be a natural consequence of articulating the knowledge explicitly, as long as it is articulated well via good domain modeling.

But the greatest difficulties in institutionalizing reuse lie on this private-to-shared axis. Even Janice did not see the extent of this. She recognized that the reuse initiative was about cultural change, but she failed to see the depth of the required change: that it involves embracing not just the idea of reuse and a general model of reuse processes, but also the knowledge, assumptions,

conventions, expectations, and commitments that are expressed in a domain model. Until the domain model itself is shared and embraced, the cultural shift has not occurred.

We see again, then, the danger of starting a pilot project without establishing buy-in from key stakeholders to encourage shared ownership of the results and support scale-up to other client projects. To do this in a sustainable way requires significant organization-wide planning. This is evident from the diagram in Exhibit 15, which shows the inevitable tension between local project interests and common organization-wide benefits.

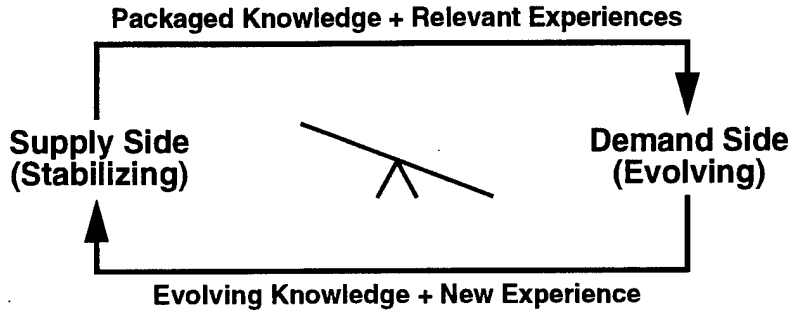


Exhibit 15. The learning cycle underlying asset supply and demand

This learning cycle is the core dynamic within all software reuse. The tension between codifying and stabilizing knowledge for reuse in a variety of contexts and evolving and adapting that knowledge for specific application needs does not disappear with the appearance (or even the endorsement) of a domain model because the projects themselves occur in a dynamic customer environment. Reuse and evolution are two sides of the same coin. To attempt one while ignoring the other is to ensure failure.

Reflection on Resistance: "Reuse is too hard"

The problem of shared domain models, which we have seen to lie behind the difficulties in scaling up, appears in everyday practice between software developers. Consider the following very common interaction: one developer has created a component which he believes to be reusable in a variety of contexts. He hears of a colleague whose current task could make good use of such a component; so he informs the colleague of the component's existence.

The colleague attempts to use the component, and discovers the hard way (after spending significant time trying to use it) that it does not fit into the design scheme — the network of assumptions, conventions, and commitments — upon which his application is based. This scheme cannot easily be changed because it involves the use of several other existing components and a code generator, all of which conform to the same design assumptions. As a result, the second developer eventually decides that he cannot reuse the first developer's component. He feels that he has wasted precious time trying to do so, and this reinforces his tacit belief that reusing other people's software (except for well established commercial products) is not effective.

From this example we draw the following lessons:

- *Failure to articulate architectural or design assumptions can lead to reuse failure. (Knowledge must progress from implicit to explicit.)*
- *Failure to negotiate architectural and design assumptions can lead to reuse failure. (Knowledge must progress from private to shared.)*

- *Failed attempts at reuse reinforce skepticism. (Retreat back to private knowledge, or at best, when it occurs, implicit shared knowledge.)*

Event 18: LoB Manager Fails to Champion Asset Base

The LoB manager declines to exert any influence on the projects under him to adopt (or try to adopt) the asset base. This confirms the indications during the presentation, in Act Two, that his support for the initiative is lukewarm. Since the pilot domain analysis was not placed in the context of a longer term plan, it is simple to let its results drop out of the limelight now. There is no perceived need to evaluate the reasons for non-adoption, to see whether the asset base or the underlying domain model can be adapted to meet the needs of other projects, or to draw lessons about the pilot project itself: its level of funding, scope, method, staffing, etc.

To the LoB manager, the experience falls under the category of failed technology experiments. The monetary loss is not too serious because the level of funding was kept down, precisely because of this risk. The larger issues, about what the experience means for the organization's business processes and software development competency, are not paramount for the LoB manager because he never fully understood what the initiative was about. His apparent fickleness is just a consequence of the gap in belief models which the presentation did not succeed in bridging.

Again, we see how the organization's paradigm has not shifted from producing products to producing value through knowledge and learning. In a world in which this shift was truly embraced, we would expect the LoB manager to have a different view of his job than just, "Where shall I invest the money?" Rather, he would be a manager of learning, and would champion a process for capturing the learning to make better future investments, even if he still chose not to champion the asset base so that other projects would not have pulled out.

Until such a time, however, the team will have to settle for doing do their own learning and conclude, as stated above, that their presentation failed to bridge the belief gap. In retrospect, the team not only needed to anticipate this consequence, but perhaps needed to use the Ladder of Inquiry and belief mapping in more individual and informal conversations over time with the LoB manager.

Event 19: Corporate Top-Down Initiative Launched

The appearance of the "gladiators" — a corporate SWAT team whose mission is to bring reuse to the LoB — is the next swing of the technology seesaw that occurs in the absence of long-term commitment (see Exhibit 16).

The good news is that reuse is (apparently) being taken seriously at the top levels of the company. The bad news is that the SWAT team, too, is likely to view reuse as the next latest technology, to be disseminated through conventional technology transfer mechanisms (promotion, training, incentives). If so, it is reasonable to expect a replay of much the same scenario as the pilot domain analysis, perhaps on a larger scale.

Perhaps, however, the SWAT team understands the idea of reuse as technology transfer in reverse, the principle of knowledge creation as value creation, the view of evolutionary domain modeling as a form of team learning. If so, when they enter the scene they may take some time to consider the lessons of the scenario we have just witnessed; they may work with the local team to identify the disconnects that account for the pilot's less-than-resounding success; they may even take those lessons with them on their subsequent stops along the company's software development map.

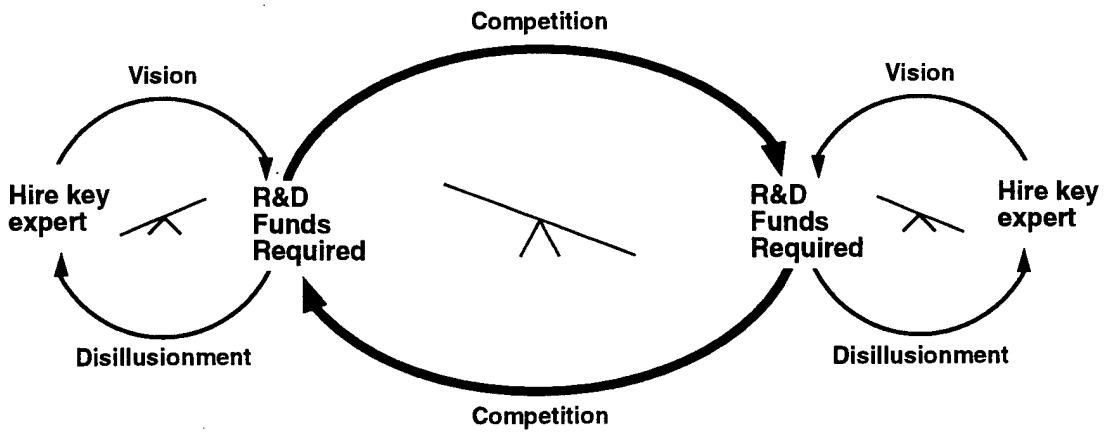


Exhibit 16. Technology investment seesaw

If the SWAT team does promote the concepts and tools inherent in the LIBRA approach presented in this document, their efforts may, over time, engender new learning-oriented, reuse-supportive patterns of interaction with the organization. The next section offers examples of some patterns of interaction that are characteristic of learning-oriented reuse.

5.0 Reuse as a Network of Interactions

Preceding sections of this document have illustrated how dramatic scenarios, system diagrams, belief maps, and the Ladder of Inquiry can be used to diagnose the current state of reuse within an organization. These techniques help make visible patterns of behavior and belief that lead to breakdowns or missed opportunities. The tools and techniques presented so far have been diagnostic or descriptive, in that they could be used to describe any interactions, both those supporting and those inhibiting systematic reuse. However, they do not reveal much about patterns that are characteristic of the desired end-state: an organization that has made the transition to systematic, learning-oriented reuse in its software engineering practices.

This section describes some archetypal patterns that can help you to identify opportunities and objectives for learning-oriented reuse within your organization. Inquiry-based assessment helps you see the current state more clearly; the patterns we present here create a bridge from where your organization is now to a desired future vision. The creative tension between current and possible future states enhances the motivation for change. The reuse proponent can use the network archetypes presented in this section as a "lens" to scan current organizational structures and interaction patterns to discover potential reuse-supportive networks not yet fully realized in the organization. In Section 6, we will present ways of identifying specific steps to take to help create and sustain reuse-based interactions.

5.1 The Reuseful Organization

We saw in the case study how the success of a reuse initiative depends as much on the nature of the interactions between stakeholders:

- their overlapping or conflicting agendas
- the thoughts that remain unspoken
- the shared or divergent assumptions and beliefs

as it does on the acceptance of reuse as A Good Thing. Reuse is technology transfer in reverse: taking what the best software engineers already do, and extending it along both axes of the Knowledge Evolution Grid (see Exhibit 1). The two axes of the grid pose two broad challenges:

- Articulating reusable knowledge
- Sharing, negotiating, and refining that knowledge into shared assets

To the extent that the interactions between stakeholders do both of these, precisely to that extent is technology transfer in reverse occurring, and to that extent is the organization moving towards institutionalized reuse (in fact, it is practicing reuse). We call this a "reuseful" organization.

The defining properties of a reuseful organization are:

- It is knowledge creating
- It learns
- It converts its learning into knowledge assets
- It applies those knowledge assets to produce further value

In other words, a reuseful organization is one that recycles knowledge. It is characterized by a cyclical value chain in which existing knowledge (legacy) is built upon to generate new products or services, new knowledge is created in that process, the new knowledge is integrated with the previous knowledge (perhaps complementing or displacing it), and is packaged for reuse in future products and services:

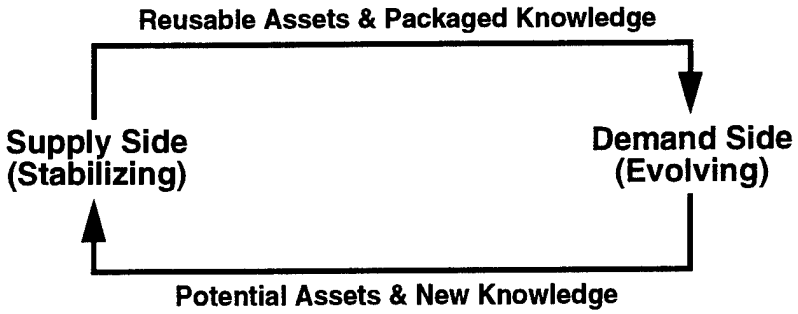


Exhibit 17. The reuseful organization recycles knowledge

The case study demonstrated how an initiative can flounder without this conceptual grounding. Let's reiterate some of those lessons:

The engineers (Joe, Mike, and June) focused on the flow from right to left. They did not fully appreciate the impact of domain scoping on the potential for scale-up and winning consensus, which are preconditions to establishing the flow from left to right. Adopting a view of software design as shared and recyclable knowledge would have enabled them to align their agenda more closely with Janice's and Andy's.

Janice, the technologist, and Andy, the marketing person, focused on the flow from left to right. They did not fully appreciate the complexity of the tradeoffs that must be considered in arriving at a common domain model. Adopting an evolutionary, learning-driven view of reuse would have enabled them to align their agenda more closely with that of the engineers, who have to deal with the bits and bytes.

Ross did not understand domain modeling, and he did not see how reuse was anything out of the ordinary. Had he considered the potential for recycling knowledge in his own activities, the obstacles to doing so, and the value of modeling as a way of articulating the content and flow of such knowledge, he might have seen analogies between his own challenges and those of the software engineers in his Line of Business. He might then have been more likely to provide needed support for the reuse initiative, and to go to bat for it when difficulties arose.

Andy did not have the technical background necessary to understand the harsh realties that Joe, June, and Mike faced, but he understood the basic idea of software reuse. He was also an expert in conveying and negotiating concepts with audiences that may not be initially receptive, finding ways of fitting these concepts into his listener's belief framework, current practices, and constraints. If Andy had been able to make the connection between the knowledge negotiation that he performed every day and the scale-up problems encountered by the reuse pilot, he might have been able to assist the pilot team in "selling" their asset base to other projects. Beyond this, he might have been instrumental in selling the idea of knowledge reuse to managers throughout the organization.

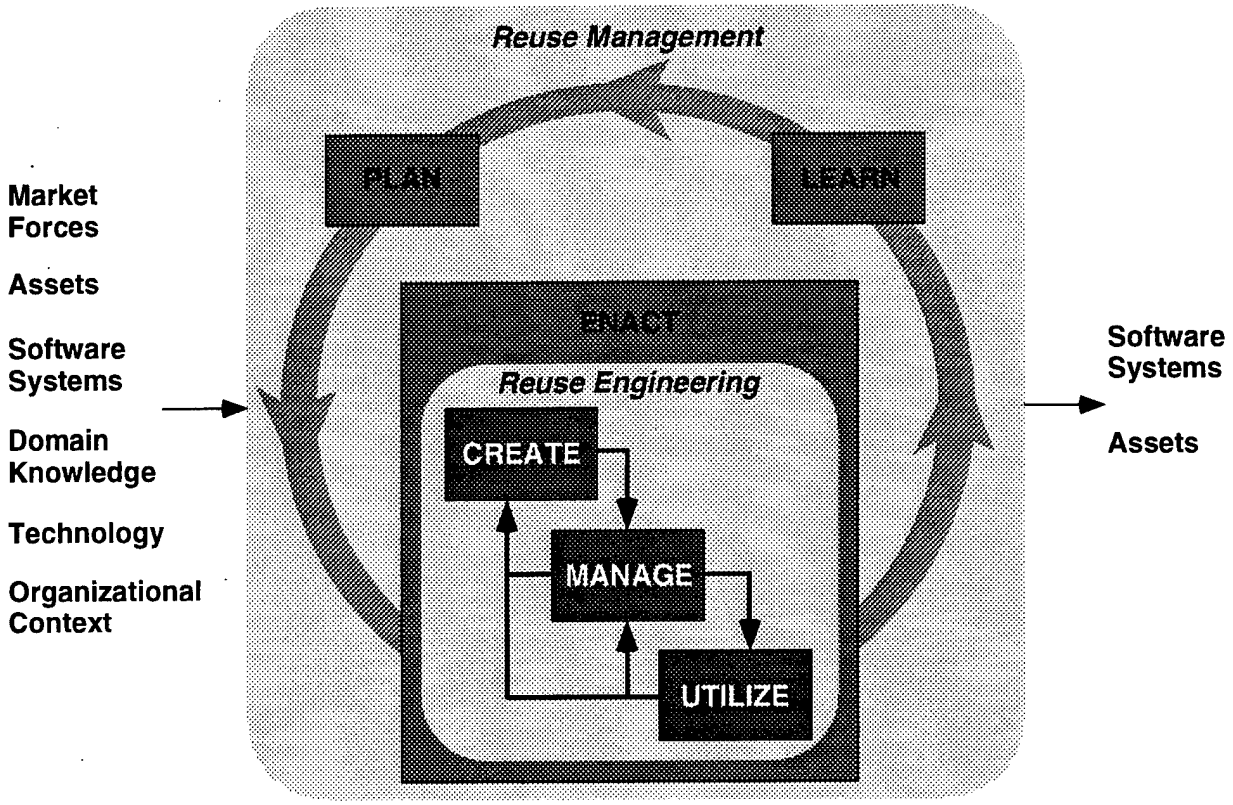


Exhibit 18. STARS Conceptual Framework for Reuse Processes

5.2 Archetype of a Reuseful Organization: The CFRP

In this section we explore the STARS Conceptual Framework for Reuse Processes (CFRP) as an aid in moving towards a reuseful organization. The CFRP comprises a set of interaction patterns that are fundamental to reuse within software organizations. These patterns are represented in the form of two interconnected "process idioms" called *Reuse Management* and *Reuse Engineering*. These encompass the management/organizational aspects and product engineering aspects of reuse, respectively. The process idioms are decomposed into "process families," and these in turn are decomposed into "process categories." The process idioms and families are shown in their most basic form in Exhibit 18.

The ***Reuse Management*** process idiom describes a cyclic pattern addressing the establishment and continual improvement of reuse activities within an organization. The Reuse Management idiom consists of the following three process families:

- ***Reuse Planning*** processes which: (a) Assess the current reuse capabilities, assets, expertise, and business context of an organization; (b) Establish specific objectives, strategies, success criteria, and metrics for the organization's reuse program; (c) Establish the scope of the reuse program by selecting the domains and application product lines the program will address; (d) Plan an interconnected set of reuse projects addressing Asset Creation, Management, and Utilization within the selected domains and product lines; and (e) Plan infrastructure capabilities to support the projects.

- ***Reuse Enactment*** processes which: (a) Manage the planned reuse projects by initiating and retiring them, monitoring and controlling their execution and interaction, and recording mea-

surements and project history; and (b) Establish and maintain a reuse infrastructure that satisfies the projects' needs.

- ***Reuse Learning*** processes which: (a) Collect and analyze measurements and project history to evaluate project performance relative to reuse program and project objectives; (b) Explore potential innovations that could have dramatic impact on the reuse program; and (c) Make recommendations to Reuse Planning processes for future enhancements to reuse capabilities, based on the project evaluations and discovered innovations. Reuse Learning processes close the feedback loop from reuse projects back to Reuse Planning.

The ***Reuse Engineering*** process idiom describes a chained pattern of activity that addresses the development and reuse of reusable assets. The idiom explicitly recognizes the value of the brokerage role in facilitating activity and interaction among producers and consumers. The Reuse Engineering idiom consists of the following three process families:

- ***Asset Creation*** processes which: (a) Produce domain models describing existing and desired capabilities in a domain; (b) Develop domain architectures to serve as frameworks for constructing systems with desired capabilities within the domain; and (c) Develop other reusable assets, spanning the life cycle, that are consistent with the domain models and architectures.

- ***Asset Management*** processes which: (a) Establish, maintain, and manage access to collections of assets (*asset bases*) for use by asset utilizers; (b) Develop models describing how the assets are organized and classified within the asset bases; (c) Acquire, accept, catalog, and certify assets to populate the asset bases; and (d) Provide asset base usage and brokering services to facilitate and promote reuse of assets.

- ***Asset Utilization*** processes which reuse assets made available via Asset Management processes. They: (a) Determine the criteria for selecting and applying assets in the context of application system requirements; (b) Identify candidate assets for evaluation; (c) Select specific assets to reuse from among the candidates; (d) Tailor the selected assets to meet application system requirements; and (e) Integrate the assets and other components to construct the desired application system.

Using the CFRP to Guide Assessment and Intervention

The CFRP patterns are useful for diagnosing where reuseful interactions are lacking or for identifying where such interactions may already be occurring:

- As a form of diagnosis, you can begin with a representation of current interactions and communication paths within your current environment. Then look for the CFRP pattern that seems to correspond most closely to the nature of the environment. Overlaying the CFRP pattern on the current description will reveal various mismatches which can be explored with questions like the following:

  — Where are the current or potential Asset Creator, Utilizer, and Manager roles in this setting?

  — What interactions must be added, damped down, or changed to complete the pattern?

  — Who could I bring together in a meeting to enable some of the new interactions needed to complete or repair the pattern?

- Alternatively, one can work from the CFRP patterns and look for situations in the current business environment that correspond to those patterns. The advantage of this technique is that the CFRP can help you identify potential networks that would not have been perceived

as related without the pattern as a guide. For example, the organizational groups most closely functioning as the "Reuse Learning" process for a given application project might be a group that would not otherwise have been considered in planning (e.g., the documentation group, the corporate standards group, the project metrics group).

Following are some examples of how the CFRP can be applied in these ways:

- Several projects within a contractor organization are working on similar types of applications for different customers. Inquiry reveals that there are no systematic learning processes in place enabling these groups to share lessons learned, best-practice techniques, components, architectures, etc. The next step would be to create a situation where such interactions could occur. This would be a catalyzing event which, if successful, could lead to ongoing learning interactions.

- A group has been set up to act as brokers for components to be reused across the organization. Each component includes the original author's name; but the recommended process is for users of the component to go back to the brokers with questions, bug reports, etc. However, because the organization is small and informal, many users of components begin to go immediately to the developer with complaints. As a result, several developers begin to get annoyed, because there is no mechanism for them to charge or bill for the time spent in these interactions (after all, they're not supposed to be happening). At the same time, the brokers lose access to some critical information, such as problems in their standard way of documenting components' required conditions for use. One next step here could be to initiate an inquiry enabling all the players to see the interactions more globally.

- A group has been tasked with porting a system across five different UNIX platforms. This is an opportunity to learn systematically between each iteration. For this learning to happen, interactions must occur between people who have been working on separate aspects of the porting task, or people who have done analogous tasks for different platforms. In the former case the people involved may have interacted during the task in terms of their workflow; but bringing them together to debrief in a new conversation allows different interactions than those dictated by the workflow. This could lead to a meeting that directly creates learning for reuse in the organization, with no top-down driven (or even visible) management-led program for systematic reuse.

Scalability. The CFRP is scalable. It is just as useful for analyzing interaction patterns at the personal level as it is within (or even across) large-scale organizations. For example, individuals can identify the lack of a Reuse Learning process (linking Enactment and Planning processes) in their immediate work environment. They can decide to be systematic about reuse in their own practice simply by asking these types of questions and searching for these types of opportunities routinely in their work.

Not everything scales, of course. If an individual's efforts are frustrated too many times by a work climate or culture that is inimical to such reuse initiatives, a tension will be created that will lead to some shift; either the person's giving up, or moving on, or propagating the ideas out to their larger network of connections.

Mapping to Organizational Levels. In a traditional, large-scale hierarchical software organization, one guideline for applying the CFRP Plan-Enact-Learn pattern is that the initiative should involve interactions among a minimum of three contiguous levels of management. If one imagines a set of independent projects as sibling Asset Utilization projects in the CFRP sense, then the corresponding Asset Management and Asset Creation processes must be coordinated from a higher level within the organization. Similarly, if a reuse project is to be treated as part of a

broader Plan-Enact-Learn loop within an overall reuse program, the program must be coordinated at a higher level within the organization.

This guideline, though somewhat hypothetical, challenges some typical assumptions made in reuse planning. For example, a reuse proponent might assume that having top-level management on board (e.g., the CEO) ensures acceptance for a reuse program. In fact, a gap of several levels between a committed CEO and a skeptical division manager could effectively curtail the chances for successful reuse. Conversely, an "island of support" with a strong, enthusiastic division manager could make progress even with initial lack of buy-in from higher in the organization.

Relation of the CFRP to the Case Study

Viewing the case study from the standpoint of the CFRP idioms, we can identify several breakdowns or missed opportunities for interactions, and areas where new interactions could have been introduced. These include:

- ***Lack of Asset Management Process***

  The absence of a clearly defined Asset Management process in the domain analysis pilot project clearly contributed to breakdowns. Joe, Mike and June were, temporarily, Asset Creators, but it was left unclear how the assets would be managed and delivered to Asset Utilizers.

  This lack of clarity may have contributed to the rift that developed between the engineers and Janice and Andy; i.e., their feeling of being "used." The plan had not really addressed what it would take to make the asset base sustainable. Janice's focus was primarily on the pilot project as demonstration, a basis for convincing management to scale up the approach within the Line of Business. Ironically, though, she paid inadequate attention to what was needed to make the initiative really work in practice.

  When problems occurred with the customer projects, there was no brokerage function to shield the asset developers from direct client project pressures. There was no one whose "stake" lay primarily in the continued and cross-project use of the assets. As a result, one of the initial client projects dropped out, no additional clients were signed up, and the pilot project was perceived as a failure.

- ***Lack of Reuse Learning Process***

  Few attempts were made in the case study to learn systematically, either from past experience or from the experience gained on the pilot project itself. Two examples of where such learning could have made a difference are:

  At the start of the initiative, leading up to management's approval of the pilot domain analysis, it would have been useful to initiate a learning step to benefit from prior reuse experience in the company, both positive and negative. Janice brought some knowledge of reuse experiences from the external community, and other members of the team researched reuse issues in preparing for the presentation. However, as evidenced by unanticipated management objections during the presentation, there was insufficient grounding in the local experience at the company.

  After the apparent failure of the pilot project (especially, its failure to scale up), another learning step could have been initiated to record the project's experiences and, in the spirit of inquiry, to analyze the reasons for the failure. Without this learning process, the failure of the effort will likely seep into the shared, informal folklore of the company and contribute to greater skepticism about the next reuse initiative.

5.3 Knowledge Creation and Evolution Roles

A reuseful environment can be viewed in terms of a number of distinctive job roles representing different ways in which people participate in reuse. We will use these roles as the basis for describing, in dialog form, a set of interactions that is characteristic of reuseful organizations. In describing the roles below, we will note who played (or could have played) what role in the scenario case study in Section 4.

The roles are generic: they do not necessarily correspond to positions in an organizational structure. Any person involved in the software engineering process may assume these roles, and the same person will typically assume different roles at different times. Our references to the case study will illustrate this flexibility.

The roles can and should occur at all levels of an organizational hierarchy. For example, in a reuseful organization, everyone is a knowledge producer at one time or another: the knowledge may take the form of a software component, or an equipment failure analysis, or a strategic plan for the company. In all of these cases, the knowledge is a potential asset for the organization.

Here is a brief description of the roles:

- ***Opportunity Perceiver.*** A person in this role is looking for and identifying opportunities to develop software by building on the organization's existing assets. Marketing people (e.g., Andy in the case study) typically perform this role in looking for external opportunities to market and deploy assets. Project planners and engineers are more likely to explore internal opportunities by continually assessing the organization's asset base and thinking of novel ways of applying or recombining what has already been developed.

  At the start of our case study, Mike was playing this role.

- ***Knowledge Investor.*** A knowledge investor is one who has at her disposal financial (or other) resources that may be applied to knowledge creation. Software development in a reuseful organization is a form of knowledge creation; but depending on the level of investment, the products of the development may or may not become part of the organization's knowledge assets. The decision to apply the resources necessary for this to happen is made by the knowledge investor.

  In the case study, Ross was the primary knowledge investor; however, the project managers under him also played this role when they decided to support or pull out of the pilot project on the basis of the project resources they had available.

- ***Knowledge Producer.*** This is the primary role of developers and engineers who participate in building a software system. Every engineering decision, every life cycle product, every problem encountered and solved, and every lesson learned is a chunk of knowledge that may be useful to others in future work. In a reuseful environment, the creation of such knowledge is viewed as the primary task of a system builder. The resulting systems are the primary manifestation of the created knowledge, but they are not the only one. Lessons learned, experience gained, and techniques discovered are other forms of knowledge produced during software development.

  Joe, Mike, and June were the principal knowledge producers in the case study. Other software engineers on the Line of Business's projects were also knowledge producers.

- ***Knowledge Evaluator.*** A person in this role is looking at the newly created knowledge that is embodied in some artifact — for example, a software component or an engineering study —

and assessing whether it represents a (potentially) valuable asset for the organization. If the answer is yes, the artifact — or another artifact that better captures the valuable knowledge — should become part of the organization's asset base. The evaluator role is necessary because the creation and maintenance of a reusable asset base does not come for free. Integrating new knowledge into the asset base requires effort: for example, to decide whether the artifact belongs in the asset base, how it relates to existing assets, whether it must be modified before including it in the asset base, whether a modification of an existing asset will suffice to represent the new knowledge, and, if modification is required, how much effort is required to perform the modifications. The knowledge investor applies the necessary resources, but she relies upon the knowledge evaluator to determine whether such investment is appropriate.

In the case study, Joe, Mike, and June were acting as knowledge evaluators when considering the organization's past and current projects as input to the domain model.

- ***Knowledge Organizer.*** A knowledge organizer is one who creates and maintains the schema by which reusable assets are organized. The knowledge organizer is tasked with maintaining the consistency, integrity, timeliness and accessibility of the asset base's organization. When new knowledge is created and the evaluator has determined that the knowledge should become a corporate asset, the organizer must then decide on the following: where to place the new knowledge, how to connect it to the assets already present, and whether the existing schema must be changed to accommodate the new knowledge.

  In the case study, Joe, Mike, and June were acting as knowledge organizers when they developed the domain model and designed the asset base.

- ***Knowledge Broker.*** The knowledge broker is a person who directs others — usually a knowledge producer or knowledge adapter — to the reusable assets they need. The role may be viewed as part of the traditional librarian function. In this setting, however, the destination is not necessarily a "place" in a repository — it may also be the names of people who can provide useful knowledge about a type of task.

  It is interesting to note that this role (related to CFRP Asset Management processes) does not appear in the case study, a consequence of the failure of the pilot to attract additional client projects once the asset base was in place.

- ***Knowledge Adapter.*** When a developer or engineer reuses an asset, he may have to adapt it to the new context into which it is placed. The asset may have been built with such adaptation in mind, e.g., parameterized according to the variables that the creators of the asset foresaw. Adaptation then consists of giving the variables appropriate values. It is possible, however, that the creators of the asset did not foresee the need for a certain type of variability, in which case the reuser might have to modify the content of the asset (for example, its design or code), remove portions of it, or add elements that adapt the asset to the new context. All of these scenarios are forms of knowledge adaptation. When performing such functions the software engineer is assuming the role of knowledge adapter.

  In the case study, software engineers who attempted to use the pilot's results were (at least potentially) playing the role of knowledge adapters.

- ***Model Negotiator.*** If incorporating new knowledge requires a significant change to the asset base schema — or to the underlying model that determines the structure of the asset base — the question arises as to whether the "old" model or the "new" one better represents the organization's knowledge and activity. A major change in the model indicates a change in the way the organization views its own work. Such change does not, and should not, come easily, and the rationale for the change must be clearly understood and tested. A dialog must occur

between the producers of the new knowledge and proponents of the existing schema. (A "proponent" of the existing schema might be the knowledge organizer, or just documented rationale.) The goal of the dialog is to discover the significance of the change and to assess whether it is really necessary. The model negotiator facilitates this dialog.

In the case study, one project had negative experiences with the assets coming out of the pilot. This could have been an opportunity for model negotiation resulting in an improved domain model and asset base.

- ***Paradigm Gadfly.*** When adaptation has repeatedly been judged as too extensive or too expensive to be justified — or the asset base contains too many exceptions to what was once regarded as a generic, tailorable asset — a paradigm shift may be in order. This is a fundamental rethinking of the nature of the organization's expertise and activity, and it occurs even in a reuseful organization because knowledge — for example, user requirements and design technique — inevitably evolves. The paradigm gadfly is the instigator and champion of this shift in organizational understanding.

  The paradigm gadfly is the first to recognize that what were previously considered exceptions have now become the norm. This causes him to question the assumptions behind the current schema and contents of the asset base. He engages in a dialog with the knowledge organizer and the model negotiator to determine whether fundamental change is necessary.

  The knowledge organizer herself may assume this role when, in the process of incorporating new knowledge into the asset base, the inconsistency or inflexibility of the current schema becomes apparent. Alternatively, a knowledge adapter may assume this role when substantial difficulties occur in the attempt to reuse an asset. Or a knowledge producer may become a paradigm gadfly when he has produced an artifact that is very different in nature from existing assets, or he has used design techniques that are radically different from those used in the past.

  In the case study there is no paradigm gadfly because reuse has not yet occurred over an extended time period. However, there is a sense in which the one project that tried to use the asset base and failed can be considered a paradigm gadfly. If the pilot domain model was a faithful representation of past practice, then the failed attempt was an indication that past practice no longer sufficed to meet current needs.

Exhibit 19 is a simplified depiction of the interactions between these roles. The arrows between roles represent statements or questions that initiate a dialog; they are not data flows.

The essential interactions shown in Exhibit 19 pertain to the creation, recognition, evaluation, and interpretation of knowledge:

- The *Opportunity Perceiver* asks the *Knowledge Broker* for insight into the organization's competencies ("What do we know?").
- The *Opportunity Perceiver* informs the *Knowledge Investor* of opportunities to capitalize on the organization's knowledge. ("An opportunity!").
- The *Knowledge Investor* provides resources to the *Knowledge Producer* to support the creation of knowledge assets ("Resources").
- The *Knowledge Producer* informs the *Knowledge Evaluator* of newly created knowledge, for potential incorporation into the organization's knowledge-asset base ("Here's something new").

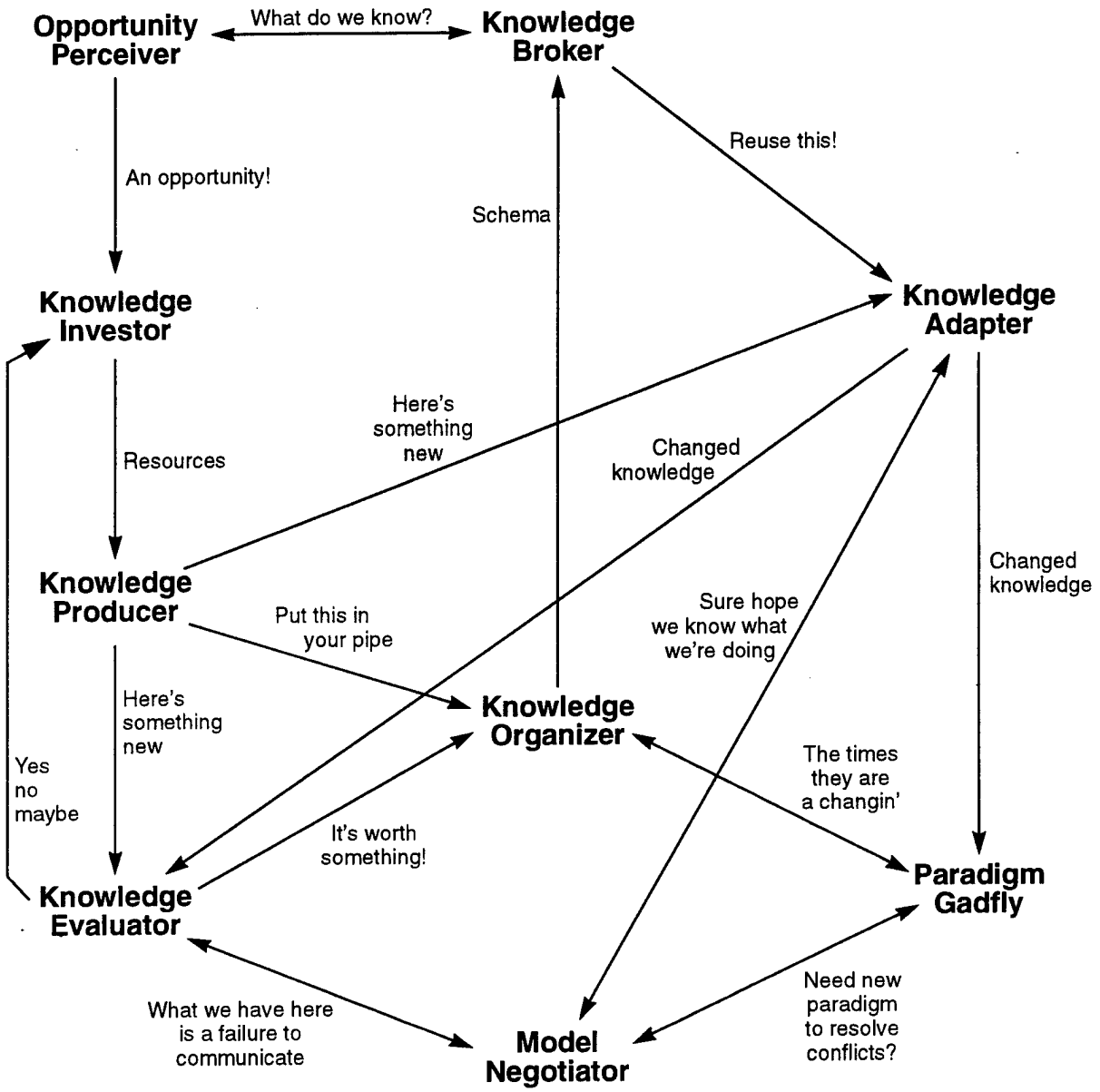


Exhibit 19. Reuseful roles and their interactions

- The *Knowledge Evaluator* advises the *Knowledge Investor* as to whether it is worth packaging a newly created piece of knowledge as an organizational asset ("Yes, no, maybe").
- The *Knowledge Producer* informs the *Knowledge Adapter* of newly created knowledge that might be of use in some activity that the latter is performing ("Here's something new").
- The *Knowledge Producer* informs the *Knowledge Organizer* of newly created knowledge, for inclusion in the organization's knowledge-asset base ("Put this in your pipe").
- The *Knowledge Evaluator* advises the *Knowledge Organizer* as to whether it is worth pack-

aging a newly created piece of knowledge as an organizational asset ("It's worth something").

- The *Knowledge Organizer* informs the *Knowledge Broker* of the schema by which the knowledge assets are organized ("Schema").
- The *Knowledge Broker* informs the *Knowledge Adapter* of potentially reusable knowledge ("Reuse this").
- The *Knowledge Adapter* discusses with the *Model Negotiator* the implications of modifying an asset ("Sure hope we know what we're doing!").
- The *Knowledge Adapter* informs the *Knowledge Evaluator* of new knowledge that was derived from previous knowledge assets ("Changed knowledge").
- The *Knowledge Adapter* informs the *Paradigm Gadfly* of new knowledge that was derived from previous knowledge assets ("Changed knowledge").
- The *Knowledge Evaluator* discusses with the *Model Negotiator* the rationale underlying past decisions about and proposed changes to the knowledge-asset base organization ("What we have here is a failure to communicate").
- The *Paradigm Gadfly* discusses with the *Model Negotiator* whether the current model of organizational knowledge is adequate in light of newly created knowledge ("Need new paradigm to resolve conflicts?").
- The *Paradigm Gadfly* discusses with the *Knowledge Organizer* the implications of changing the knowledge-asset base organization ("The times they are a-changing").

5.3.1 Dialogs Among Roles

We now present 12 types of dialog that are typical of a reuseful organization. There are many possible variations to the dialogs: some are noted in the descriptions, the rest we hope will be apparent — or, better, will be discovered as the reader engages in these patterns of interaction. The dialogs, just as the generic roles described above, can and should occur at all levels of an organization. The extent to which these dialogs are occurring — the degree to which they represent the predominant forms of interaction among members of the organization — is a measure of how "reuseful" the organization is, and of the progress made towards becoming a reuseful organization.

Each of the dialogs is triggered by an initial question. The dialogs can be grouped into categories as shown below:

1) Inquiries about the organization's reusefulness (assessment):
   a) Is there potential for sharing knowledge between activities FOO, BAR, and SNA?
   b) Are we taking advantage of our knowledge assets?
   c) Are we investing enough in our knowledge-asset base?
2) Inquiries about packaging knowledge for reuse (supply):
   a) Could the knowledge FOO, which came out of project BAR, be an asset to the organization as a whole?

b) Is it worth developing FOO for reuse?

3) Inquiries about reusing knowledge (demand):

a) Is this an opportunity to reuse asset FOO?

b) Are we reinventing capability FOO?

c) Who knows about topic BAR? (Or: What do we know about BAR?)

d) Is it worth reusing asset FOO if we have to adapt it?

e) Should we build capability FOO or "buy" it?

4) Inquiries about the validity of the knowledge base (learning):

a) What does it mean that the assets FOO and BAR seem similar but are different?

b) Is it time for a paradigm shift?

We present each dialog as a prototypical "script" played out among a subset of the roles.

Inquiries about an Organization's Reusefulness

Dialog 1a: Is there potential for sharing knowledge between activities FOO, BAR, and SNA?

This dialog occurs between Opportunity Perceivers and Knowledge Producers. It represents the first step towards recognizing knowledge as an organizational asset. It might run something like this:

Opportunity Perceiver: From the descriptions I've heard of your projects, it sounds to me as if you develop many similar capabilities. I wonder whether some economies could be obtained by pooling ideas and resources?

Knowledge Producer 1: We've recognized this for quite some time. The problem is that each project has its own unique constraints and requirements, and it would take some serious thought to come up with a way of sharing capabilities.

Knowledge Producer 2: Of course, we already do that at the level of common subroutine libraries and off-the-shelf packages. All of the Line of Business projects use the same graphical user interface toolkit unless a particular customer objects vociferously. But the real economies would come from sharing application capabilities.

Knowledge Producer 3: The problem is that when a project gets turned on, it is under intense time and budget pressure right from the start — we always bid jobs as optimistically as possible. No one ever has the time to do the thinking that Knowledge Producer 1 is talking about.

Opportunity Perceiver: But you all agree that in principle it would be a good thing?

Knowledge Producer 1: A funny thing about software is that good things in principle turn out quite often to be not so good in reality. There are so many layers of interrelated issues which one cannot see in advance.

Knowledge Producer 2: But I think we would all agree that it is worth considering: there is at least potential.

Knowledge Producer 3: But where could we get the resources even to consider it?

Opportunity Perceiver: You need to be paid to reflect, for a change. A lot could come out of it.

Relation to the Case Study

This question was posed by Mike and Joe at the very beginning, and it triggered the entire reuse initiative. It should have been posed again towards the end of the story when scaling up proved difficult. Asking the question at that point would have triggered a learning process in which Mike's initial conclusions ("Of course we can share knowledge") were contrasted with the harsh realities of the projects who declined to adopt the reusable assets: "We tried but failed," "We can't wait for them," "We're concerned about who will own and maintain them." Why did Mike's initial perception, which was supported by a seasoned engineer like Joe, turn out to conflict with the team's actual experience? In a reuseful organization, investigating the question will lead to insight that can itself become part of the organization's knowledge base.

Dialog 1b: Are we taking advantage of our knowledge assets?

This is basically a dialog between Opportunity Perceivers. It could be the reflection of a single Opportunity Perceiver.

Opportunity Perceiver 1: Competition is getting stiffer. Company FOO, Inc. has carved out a market niche as the experts in PC-based 3D animated Web moles. The whole world knows them for that and goes to them for it. If we don't define ourselves in a similarly clear fashion, our market share is going to plummet over the next year.

Opportunity Perceiver 2: If you look at our products over the last two years, you can see some distinguishing characteristics. There are certain things that we do better than FOO, Inc. — always have; we have the best people for it. But we've never capitalized on that distinction, made it a part of our stated identity.

Opportunity Perceiver 1: So we do have some unique areas of expertise. We need to turn them into corporate assets.

The dialog might also involve a Knowledge Broker:

Opportunity Perceiver: Have you noticed any sustained "best sellers" in our software repository?

Knowledge Broker: Certainly. We have an excellent collection of scheduling algorithms, for example; nobody develops their own anymore. But I've noticed that there are some equally excellent assets that nobody ever requests. I don't know whether it's because they don't know about them, or are concerned they will be too difficult to reuse. I just know that we do not reuse as much as we could.

Some consultation with Knowledge Producers and Knowledge Adapters might also occur:

Opportunity Perceiver: Jack (the Knowledge Broker) just told me that we have some excellent generic design assets that nobody ever requests. Don't you folks have a need for them?

Knowledge Producer: I honestly did not know about them.

Knowledge Adapter: I did, but frankly my one attempt to use them cost me multiple staff-weeks, and I'm really gun-shy now.

Opportunity Perceiver: Jack told me about that, but he says your comments led to an overhaul of the assets and they are much stronger for it.

Knowledge Producer: We need to do a better job of checking before we build something from scratch.

Opportunity Perceiver: It's not just a matter of increasing productivity, or even increasing quality. In this competitive climate, it's a question of creating an organizational identity and culture so that the market knows who we are, what we're the best at.

Knowledge Adapter: I never thought of it that way: I'm always so immersed in the crises of the current project...

Relation to the Case Study

This dialog occurs only implicitly in the case study, in the Stirrings scene. Mike instinctively feels that his organization is not taking advantage of its assets. The glaring question is: why is Mike the only person asking this question and initiating this dialog? A major step forward in the Line of Business's reusefulness could occur if Ross and every manager under him regularly posed this question and engaged in this type of dialog.

Dialog 1c: Are we investing enough in our knowledge-asset base?

This dialog is an immediate corollary to the previous one if the answer there was No. If the organization is not taking advantage of its knowledge assets, one obvious reason may be inadequate investment. The dialog might also occur in response to a negative reuse experience, such as one project's experience towards the end of the case study.

The dialog occurs between the Knowledge Investor and someone else with a stake in the knowledge-asset base. The other participant might be an Opportunity Perceiver, a Knowledge Adapter, a Knowledge Organizer, a Model Negotiator, or a Paradigm Gadfly.

The investor may ask a host of questions:

Knowledge Investor: Do we have all the assets we want? Are they of sufficient quality? Are they flexible enough to support our projects? Are they known to developers, project managers, others? Are they ossifying through lack of use? Through lack of maintenance?

If the answer to any of the questions indicates a problem, the dialog examines the facts of the problem and seeks to explain the facts:

Opportunity Perceiver: We need a richer asset base to enable sharing of knowledge among projects.

Knowledge Adapter 1: My experience using these assets has been mixed. I no longer assume that all of their relevant features and constraints have been documented.

Knowledge Adapter 2: There are implicit architectural and design assumptions that reflect the historical source of the assets. I have run into conflicts between these assumptions and those of the contractors with whose software we have to interface.

Knowledge Producer: I might have used some of these assets on my last project if I had known about them.

Model Negotiator: These assets no longer reflect current design practice. The field has advanced rapidly and our support of the assets has lagged.

In the spirit of inquiry, each of these responses warrants elaboration. With the facts thus explained, the participants can assess whether increased investment will ameliorate the problem or whether some other factor (such as resistance) is the key.

Relation to the Case Study

No one in the case study except Mike and perhaps Janice ask this question on a regular basis. The dialog shows the Ladder of Inquiry being used to arrive at the facts underlying a problem. Had such an inquiry occurred in the case study when the scale up problems were encountered — or even before Ross decided to fund the small pilot — more reasoned steps might have been taken towards a successful initiative.

Inquiries about Packaging Knowledge for Reuse

Dialog 2a: Could the knowledge FOO, which came out of project BAR, be an asset to the organization as a whole?

In this dialog, a Knowledge Producer has presented the Knowledge Evaluator with a new artifact for inclusion in the organization's asset base. It is the task of the Evaluator to decide whether it should be included — and whether the artifact must be repackaged, or otherwise modified or enhanced, before it can be considered an organizational asset.

Knowledge Producer: I just developed this incredibly cool capability. Let me demonstrate it for you.

He does so. (Or, he supplies the Evaluator with a package of information about the new software, or whatever kind of artifact it is.)

Knowledge Evaluator: Impressive. I don't think we have anything else like that in our asset base.

Knowledge Producer: I'm sure we don't — I looked hard for one before building this.

Knowledge Evaluator: But it probably fits into category SNA in the asset base, alongside assets like ZEM and ROG.

The Evaluator is addressing an issue that is mainly the purview of the Knowledge Organizer. She has to do so, however, in order to assess the potential value of the asset.

Knowledge Producer: I agree, that's where it should probably reside.

Knowledge Evaluator: How customized is it to your specific project's needs and constraints?

Knowledge Producer: There are a couple of project-specific features. I tried to isolate them in modules with a well-defined interface so that other projects could replace them.

Knowledge Evaluator: Do you have a design narrative to clarify for me which aspects of the software you regard as generic and which are project-specific?

Knowledge Producer: Sure do. I learned the hard way that it's the only way you'll let something into the asset base — and with our new incentive program, I'm pretty motivated to get things in there.

Knowledge Evaluator: And to have them reused. That's a condition of the incentive program. It might seem as if I'm being picky but it's in your interest as well as everyone else's.

Knowledge Producer: Understood. When you look at the package I gave you, you'll see that I used Literate Programming techniques to create a nauseatingly complete record of my design decisions and rationales. It actually made development easier for me: I didn't have to hold as much in my head all the time.

Knowledge Evaluator: And did you use any of the common design patterns that are becoming part of the community knowledge these days?

Knowledge Producer: Yes, and when I departed from them I explained my decision in gory detail in the design narrative.

Knowledge Evaluator: It sounds as if we might have a valuable addition to our asset base here. Let me study the material you've given me and get back to you.

An alternative outcome of this dialog is that the Evaluator sees the potential value, but does not believe that the artifact is reusable in its current form. She would then have to decide — along with the Knowledge Investor — whether it is worth investing in making the artifact reusable; or whether — despite its project-specific implementation — the artifact would be useful as an example for projects that need a similar capability. In the latter case, the design process could be reused even if the software itself could not.

Relation to the Case Study

A similar question — "Could this existing knowledge be an asset?" — was presumably posed by the reuse team in the early phases of their domain analysis. They had to examine the existing software of the Line of Business, looking for common requirements and implementation approaches. However, the question should also have been posed towards the end of the story, when scale up problems were encountered. At that point the question would have been, "How can this new capability, developed by a particular project independently of our reuse initiative, augment and enhance our results?" In this form, the question might have triggered an evolution of the asset base that could have increased the chances of future projects buying in.

Dialog 2b: Is it worth developing FOO for reuse?

This dialog occurs in the course of a project when a capability the project needs to develop is perceived as being of potential value to other projects too. The inquiry could also occur during development of a reusable asset base, when choices have to be made about which of the many assets that could be developed, should be developed. Our example will assume the first context. In this case, the Knowledge Investor might be the Project Manager, the Knowledge Producer might be a

software engineer working on the project, and the Opportunity Perceiver and Model negotiator might be a representatives from the organization's Software Engineering Process Group.

Opportunity Perceiver: You know, there are other projects that could use this asset if you make its interface general.

Knowledge Producer: We don't need that generality for this project, and it will cost us to achieve it.

Model Negotiator: I can provide some idea of the level of generality required to meet the needs of the other projects.

Knowledge Investor: Would they be willing to chip in some funds?

Opportunity Perceiver: They might, if they had a say in how the asset was developed.

Model Negotiator: We're talking about a continuing domain engineering activity here. I would be careful about doing this under the aegis of one project.

Knowledge Producer: I have a tight schedule commitment. If we are talking about convening a domain analysis team to specify the requirements, I don't have much confidence about being able to meet my commitment.

Model Negotiator: Then perhaps I was wrong. Maybe I can give you an overall idea of what other projects need, and you can just see what you can do.

Knowledge Investor: With the understanding that the project commitment comes first, given that we're paying for it.

Knowledge Producer: So the design/cost tradeoffs will be left to me? I'm comfortable with that if you are.

Model Negotiator: Perhaps we could be involved in the design review. That way you would gain the benefit of insights from other projects, without being constrained in a way that would jeopardize your current commitments.

Knowledge Investor: It's a good idea: input from the outside generally keeps us all honest anyway!

Of course this is not the only possible legitimate outcome. If the schedule commitments were not as tight, a multi-project effort might have been feasible. There might have been an asset-base sustaining engineering activity already in place (funded by internal R&D dollars) to respond to newly discovered needs or opportunities. Or the project pressures might have been so great that any attention to other projects' needs was, in this case, simply out of the question. The dialog allows these paths to be explored in the spirit of inquiry.

Relation to the Case Study

This dialog is an example of how ongoing domain engineering can be driven by evolving project requirements; and how reusability is not an all-or-nothing proposition. In the case study, we saw how the opportunity for project-driven evolution of the domain model was lost. The choice by the Line of Business's projects to develop their own software rather than use the proffered assets was interpreted as a failure of the pilot. The dialog shows, in contrast, how it might have been

approached as an opportunity to extend or refine the asset base; or, at least, enrich the domain model with variations not previously recognized.

Inquiries about Reusing Knowledge

Dialog 3a: Is this an opportunity to reuse asset FOO?

This dialog occurs between a developer who, while building (designing or implementing) a system, is faced with a functional requirement that she suspects has been encountered before. The developer plays the role of a Knowledge Producer (the produced knowledge being the system she is building), and she converses in this role with a Knowledge Broker — someone she thinks may know of an asset that could fit the bill. The dialog begins with the basic reuse interaction:

Knowledge Producer: I need an asset that performs the function BAR, and it is subject to the constraints SNA and ZEM.

Knowledge Broker: Asset FOO is probably the closest thing to that.

What follows is a conversation about the "3Cs:" Concept, Context, and Content [Lato90]. The concept is the functional requirement that asset FOO implements, and the Producer and Broker compare this to the functional requirement that needs to be met, e.g.,:

Knowledge Producer: From the description of this asset it looks similar to what I need, but I'm not sure it's an exact fit.

Knowledge Broker: That may be because of the terminology the creator of this asset used. Let's look more closely at the description.

A close examination of the asset's documentation indicates that it might fit the bill, but the Producer is still wary. The second "C" — context — is addressed as a way of gaining more insight.

Knowledge Producer: What sorts of contexts did the developer of this asset intend it to be used in?

Knowledge Broker: I wish he had provided that information, but all we have is some experience reports of previous uses of the asset.

Knowledge Producer: Let's look at them, maybe they will give me a feel for what the asset is good for.

They examine the experience reports. These include reports of usage similar to what the Producer is intending, but there were apparently problems encountered when using the asset this way.

Knowledge Producer: It looks like they were able to use the asset, but it wasn't simply a matter of plug and play.

Knowledge Broker: No, but they did develop workarounds to the problems — maybe you could reuse those too.

Knowledge Producer: I would want to see how they implemented the workarounds. Performance is a key success factor for the system I'm building. I want to make sure they didn't insert code that will slow the system down too much.

Knowledge Broker: Luckily they included the content of the workarounds with the experience report. Take a look at it and let me know what you want to do.

The Producer now examines the content — i.e., the implementation — of the workaround, which from her point of view is part of the asset she is considering reusing. It looks satisfactory from a performance standpoint, but the implementation will clearly use a lot of memory.

Knowledge Producer: I will have to talk to the systems engineers, and maybe even the customer, about the memory implications. I'd like to reuse this asset — it will save me a hunk of development time — but I will need to get their approval. It might mean compromising on some other requirements.

Knowledge Broker: Let me know.

The conversation then triggers several additional conversations between the Knowledge Producer and her system engineering colleagues (also playing the Knowledge Producer role) as well as with the customer. Various alternatives are considered, including relaxing some of the system memory requirements, as well as the possibility of modifying asset FOO or modifying the current design so that there is more memory available for asset FOO. In the end, the developer decides that it is worth modifying her design to enable her to reuse asset FOO, and this decision becomes part of the design record for the system. She also informs the Model Negotiator of this experience because it may indicate a recurrent need for a variant of asset FOO.

Relation to the Case Study

The dialog represents what ideally would have occurred in the Line of Business's projects once the pilot asset base was developed. In the dialog we see the Knowledge Producer express some of the same concerns that caused the projects to reject the asset base; the difference being that in this dialog, the concerns are subjected to a process of discovery rather than treated as show stoppers, as they were in the case study. This is an illustration of the Ladder of Inquiry at work.

Dialog 3b: Are we Reinventing Capability FOO?

This dialog occurs when a Knowledge Producer is assigned a development task and she suspects that it may have been done before. The dialog may be with other Knowledge Producers because of their experience, but to the extent that they are responding to this question they are assuming the role of Knowledge Broker.

Knowledge Producer: This development item strikes me as something that other people must have done already. I wonder if there is anything available for me to use.

The Knowledge Producer characterizes the development item in terms of the "3Cs" — primarily the Concept (the first "C") which is a description of the purpose or function of the item.

Knowledge Broker: There is a lot of work being done in that area. You might be able to find a commercially available product, or maybe even shareware, that does what you need.

Knowledge Producer: I know that this field is moving rapidly: last year I would probably have had to develop it myself, but I suspect by now there are a lot of implementations. The problem is, how do I find out what is available?

Knowledge Broker: Our asset base has links into the World Wide Web. For this type of item you might look at the shareware repositories at URLs www.BAR.net and www.SNA.net. You might also post a request for information on the news groups comp.ZEM and comp.ROG. I also recommend contacting a couple of companies, QWE, Inc. and RTY, Inc., and there are some recent books by Briar, Patchen, and Thorn that talk about this type of technology and might mention specific software packages. Also, give Patrick MacMullen over in Space Systems a call, he might be able to give you a dump on what's out there and what really works.

Knowledge Producer: I'll follow up on those leads.

Relation to the Case Study

Like the previous dialog, this one might have led the Line of Business projects to assets developed by the pilot effort. The reluctance of the projects to make use of the pilot asset base does not necessarily indicate a reluctance to look elsewhere, e.g., on the World Wide Web. But their particular concerns do suggest the Not Invented Here syndrome, which would apply equally (perhaps more) to searching sources outside the organization. Conversely, a project culture that fostered such searches might have encouraged more open inquiry about the pilot asset base.

Dialog 3c: Who knows about topic BAR? (Or: What do we know about BAR?)

This dialog occurs when a developer — in the Knowledge Producer role — is faced with a task in an area he is not familiar with or experienced in, or a problem he is not sure how to solve. The dialog is really a series of chained references to people or assets representing expertise (or at least experience) in the unfamiliar area. The dialog begins with a statement of need by the knowledge producer, directed to a Knowledge Broker:

Knowledge Producer: I need to know what we have done in area BAR.

Or:

Knowledge Producer: I have to build a component (or system) in domain BAR and I have never done this before. Where can I obtain insight into the do's and don'ts?

Or:

Knowledge Producer: I've encountered the problem SNA in building my BAR system. Has anyone encountered this before, and if so what did they do and what did they learn?

Knowledge Broker: There is a portion of our asset base that contains systems of the type you are working on. You could use those as a model, or look at the experience reports to see what the main issues are.

Alternatively:

Knowledge Broker: I know that June has done some work in that area (or: has encountered similar problems) — why don't you contact her?

Each resource to which the Knowledge Producer is referred may in turn refer him to other resources (people or assets). In an asset base this may take the form of a series of hyperlinks

which the Knowledge Producer can traverse until he finds the information he needs. Alternatively (or complementing this), a succession of contacts with people resources may occur:

Knowledge Producer: I heard from Travis (Knowledge Broker) that you've worked in the area I'm working in right now (or: I'm encountering the problem SNA and I heard that you have successfully addressed such problems). What can you tell me to help me expedite the task?

The person being contacted for help is also a Knowledge Producer, but may in this case assume the role of another Knowledge Broker:

Knowledge Broker 2: What I encountered was a little different from what you're describing. I suppose Travis (Broker 1) referred you to me because of my work on the ROG system. I can tell you about that, but I think Henrietta (another engineer) has experience more in line with what you need.

Knowledge Producer: Tell me about your experience — it might give me some context in which to understand my own problem even if it doesn't directly apply. I'll also contact Henrietta as you suggest and see what she has to say.

Dialog 3d: Is it worth reusing FOO if we have to adapt it (or adapt our design to accommodate it)?

It is a fortunate developer who finds exactly what he needs in an existing product, in a form that fits perfectly with the rest of the system being built. In many — perhaps most — cases, some compromise and/or adaptation must occur. The question initiating this dialog indicates two ways for adaptation to occur. The reusable asset itself may be adapted, producing either a variant or an enhanced version of the asset (if a consensus of all who hold a stake in the asset can be reached). But it is equally likely that the context into which the asset will be placed — the relevant system design decisions — will be adapted to accommodate reuse of the asset.

The dialog occurs between the Knowledge Producers responsible for making such design decisions. It may also involve other roles: A Knowledge Investor's concern is to amortize previous investment in reusable assets. A Knowledge Broker helps the Producers access the information they need to make a sound decision. A Knowledge Evaluator is concerned with maintaining a coherent development approach throughout the organization and preventing spurious variation among products.

Knowledge Producer 1: It's hard to estimate what it's going to cost to adapt this asset for our needs. In my experience there are always little "gotcha's" that you can never predict until you cut the code — especially since none of us is familiar with the asset's implementation.

Producer 1 has raised the issue of the third "C" — Content. Because our ability to specify a component's behavior is limited, there is always the possibility that the "black box" description of a component (its Concept) omits crucial — often assumed — information.

Knowledge Investor: All we can do is factor that risk in as a cost element, and make our best judgment.

Knowledge Producer 2: We can also look at previous experience reusing this asset.

Knowledge Investor: And use our previous experience developing this asset to help estimate what it will cost to redevelop a similar capability.

Knowledge Producer 1: Do we have all of that information?

Knowledge Broker: Some of it, at least. The Knowledge Organizer — and the Evaluator too — are becoming more hard-nosed about requiring it when anything is added to our asset base.

Knowledge Producer 1: The problem with considering previous uses of the asset is that they may not have been subject to the same constraints as this project. I can only repeat that we should be very careful. If it were an exact fit I wouldn't have any problem, but once we start talking about modifying a component that someone else has produced — or retracting some of our own design decisions — I start to get worried. I've been burned too many times.

Knowledge Producer 2: I agree, but we should at least make an informed decision. If we decide not to reuse it, we should be able to justify that decision.

Knowledge Evaluator: Even a negative decision on your part will be a contribution, if you provide a solid rationale. It will give me more insight into what makes an asset truly reusable. It might help me make more intelligent investment recommendations.

Knowledge Producer 1: Let's develop a table of pro's and con's, and cost estimates for both approaches. An expected cost saving is a "pro," and every risk is a "con." Then we'll use that old faithful technique, engineering judgment, to decide which way to go.

Knowledge Producer 2: As we proceed we'll keep track of whether our predictions have proved accurate, and we'll add that information to the design record.

Knowledge Organizer: If your experience proves quite different from past projects, I'll have to figure out why and then modify our classification schema to reflect the difference.

Relation to the Case Study

This question should have been posed by the projects that considered using the results of the pilot — and perhaps it was. However, the resulting insights should have been fed back into the domain model and asset base, and this did not happen. Integrating the answers into the domain model would have enriched the level of decision support that could, in the future, be provided to potential asset base users. By incorporating the insights behind even negative decisions, it might also have given the non-reusing projects a stake in the asset base. In the long run, this would have helped to scale up both the interest in and usefulness of the asset base.

Dialog 3e: Should we build Capability FOO or "buy" it?

This is a continuation of Dialog 3c. The Knowledge Producer has followed up the references she received from the Knowledge Broker, and has concluded that there are several similar components available off-the- shelf. However, the issue of Context (the second "C") now looms:

Knowledge Producer: My concern is that I'm developing in a workstation environment and a lot of the commercially available components are for the personal computer market.

Knowledge Broker: Are you committed to a workstation environment?

Knowledge Producer: It would be a major decision to change that. We would have to consider whether there would be sufficient cost savings from using an off-the-shelf product. But it does make me wonder whether a personal computer would be a more appropriate platform. It might allow us to take advantage of a lot of outside work.

Knowledge Broker: Developing this type of component can easily become a work sink — there are a lot of bells and whistles that you might decide you need, and it may end up being a larger development job than you've planned for.

Knowledge Producer: It reminds me of our BAR component: when we started developing that, three years ago, there was nothing comparable around — at least nothing publicized. By the time we had invested two years worth of effort it was still in the form of an evolving prototype, but there were a bunch of commercially available products that did similar things, and better. I don't want to make that mistake again.

Knowledge Broker: Sometimes it can't be helped — you need the capability when you need it. But it is worth making that decision carefully. However you decide, be sure to document the decision and the rationale for it, and put that information in our corporate knowledge base. That way, even if the decision turns out to be wrong, at least others will be able to learn from it.

At this point a Knowledge Adapter and the Knowledge Investor may chime in with the following advice:

Knowledge Adapter: Even if there isn't anything available now, try to design your system so that you can throw out your own implementation and plug in an off-the-shelf version if one eventually shows up. That way you'll be able to take advantage of it when it appears, and you won't be playing catch-up with your home-grown implementation.

Knowledge Investor: In some circles it's heresy to talk about throwing something out after we've invested so many dollars in it. But it would be wrong to view that as a waste, in this kind of situation. I see it as a set of coordinated decisions that balance short- term and long-term needs.

This dialog applies also to situations in which the available software is an asset of the organization, rather than a commercial product. There is one major difference: since commercial products are invested in heavily and evolve quickly, they present a significant threat to outpace any home-grown development. Internally supported assets may or may not present a similar "threat." In both cases, however, the decision must be made on the basis of tradeoffs such as those considered in the previous dialog.

Relation to the Case Study

Two themes in this dialog are relevant to the case study: 1) the high cost of creating and maintaining components reliable enough to be "sold" to others, and 2) the need to view software develop-

ment as a process of successive approximations. Understanding the first point might have prevented unrealistic expectations of a small pilot effort. Understanding the second point would have softened the all or nothing, do or die criteria by which the pilot was judged.

Inquiries about the Validity of the Knowledge Base

Dialog 4a: What does it mean that assets FOO and BAR seem similar but are different?

This dialog occurs between the Model Negotiator and the Knowledge Organizer. It is an attempt to discover the significant features that distinguish, at some level of detail, assets which from a more abstract point of view are similar.

Knowledge Organizer: I've just been handed asset FOO by the Knowledge Evaluator for inclusion in our asset base, but I'm not sure what to do with it. It seems to be another version of asset BAR, but I don't fully understand their differences.

Model Negotiator: Are you sure they are different?

Knowledge Organizer: The documentation suggests that the assets behave differently. The problem is that the differences I see concern very low- level properties. They seem like implementation accidents — choices that the implementors made for convenience, perhaps. But if I just place them both in the same bin. and not provide any guidance on how to choose between them, I know I will get some angry reusers who have chosen the wrong one.

Model Negotiator: If you are correct in predicting that there is a right one and a wrong one for different contexts, then the differences must be significant.

Knowledge Organizer: Yes, I'm sure they are, but I'm not sure that the developers themselves recognized the significance. They certainly haven't expressed it in their documentation.

Model Negotiator: Then we will have to get with them and jointly try to understand the difference.

Knowledge Organizer: And if it turns out that these were accidental features?

Model Negotiator: Then we have to infer what the significant differences in behavior are going to be, and use those as the distinguishing features in the asset base.

Relation to the Case Study

This conversation might have occurred in the case study at the point where one project tried to use the assets produced by the reuse team, failed, and proceeded to develop their own. The project might then have provided the new assets to the reuse team for incorporation in the asset base. The question would trigger a learning process in which assumptions of the domain model were examined and, if need be, modified.

Dialog 4b: Is it time for a paradigm shift?

This dialog occurs when some threshold of exceptions has been reached in incorporating new knowledge into the asset base. Exceptions occur when assets do not fit neatly into the taxonomy

of the asset base, or they violate the architectural assumptions of the asset base. The dialog occurs between the Paradigm Gadfly, the Model Negotiator, the Knowledge Organizer, and the Knowledge Broker. In this example we focus on exceptions caused by evolution from a centralized to a distributed application architecture, but the architectural issue could be anything.

Paradigm Gadfly: The centralized processing architecture that your asset base assumes is completely out of date. The old systems are being replaced one by one with distributed workstation networks. Soon there won't be any use at all for this asset base if you don't seriously rethink its contents.

Model Negotiator: Many of the assets are compatible with the newer distributed architectures.

Paradigm Gadfly: Yes, but how does a potential reuse ascertain that?

Knowledge Organizer: (To the Knowledge Broker) Have you found that demand for these assets is decreasing?

Knowledge Broker: Yes, and I'm continually being asked whether the assets are compatible with a distributed architecture.

Knowledge Organizer: I've tried to include variants that have come from the newer projects. The asset taxonomy is becoming hopelessly complex because they do not really fit with the old way of doing things.

Paradigm Gadfly: We need to revisit the domain model, in a major way.

Model Negotiator: That will cost, as you know.

Knowledge Broker: But we're rapidly slipping back to a situation where we don't have an active asset base. Do we really need to go through all that again?

Model Negotiator: If there is consensus among us,. then we'll have to approach the Knowledge Investor and tell him the news.

An alternative outcome might be this: the Model Negotiator is able to convince the Paradigm Gadfly and the Knowledge Organizer that the exceptions are not as unmanageable as they believed. Instead of radically revising the domain model, then, the participants would revise the way they interrelate assets of the new and old architectures.

Relation to the Case Study

There is only a weak relation of this dialog to the case study because it assumes that a domain model has already been established and used over an extended period. Nevertheless, like the previous dialog, it indicates how domain modeling is an ongoing process in which the usefulness of the current paradigm must be continually reevaluated. Had this view been articulated and agreed upon among by the reuse team before the presentation to Ross, they might have insisted that the pilot effort be placed in the context of a longer term plan. Instead of resulting in an aborted effort, then, the difficulties encountered by the pilot would have triggered a learning process. Decisions embodied by the pilot domain model would be questioned in light of feedback provided by the projects, and the model would evolve accordingly.

5.4 Different Theories—Different Networks

In the course of collaborating on this document, the authors wrestled with two different theories about the relation between transitional and end-state reuse processes. One theory holds that large-scale changes can be introduced through a pattern of small-scale learning-based interactions. The other theory posits that organizational restructuring will be required for certain types of transformations; i.e., not all change can be incremental.

In an organization where systematic learning is very foreign to the current culture, one person practicing inquiry skills in their work is not likely to start the "chain reaction" necessary to create a large-scale shift or transformation in the organization as a whole. On the other hand, not every individual needs to make this "conversion" simultaneously. We hypothesize (or rather, until people have tried it, speculate!) that if enough people within a given network (or potential network) make the shift, the network as a whole will begin to function according to a reuse-based process. The key question is: how many is enough? Presumably at some point the reuse-supportive network of interactions would be robust enough to survive periodic disruptions from "backsliding" individuals or external environmental pressures (though clearly, some network patterns will prove to be more robust than others). How do we create the critical mass within the organization for this cross-over point to be reached?

We invite readers to find their own place within this belief map. There is not necessarily only one "right" pattern for reuse, and in fact it can be helpful to explore different patterns. There may be patterns particularly suited to your organization's business model, software development methodology and life cycle management paradigm, and lines of business. Some patterns may emphasize incremental, evolutionary learning while others emphasize larger-scale interventions like domain engineering projects or new organizational structures. The common ground underlying all network archetypes is the idea of the "network of interactions as hero": the idea, that is, that "reuse happens" as a product of the network activity as a whole. A reuse proponent's task is to bring together the right people to spark new interactions that can lead eventually to a new and sustainable network of reuse-based interactions.

6.0 Fostering Reuseful Interactions

The LIBRA approach to reuse assessment involves building a picture of the organization's current state and mapping it against the network archetypes of a reuseful organization. This leads to identifying new interactions that have the potential to move the organization towards systematic learning-oriented reuse. In this section we present additional inquiry-based concepts and techniques to aid the reuse proponent in this process.

- In Section 6.1, we present several key ideas, including the concept of "new conversations" that can be initiated and facilitated by reuse proponents as a way of shifting the organization towards a new pattern of interactions. The focus on conversations provides a setting for directly practicing inquiry-based interaction skills.
- In Section 6.2, we revisit the concept of the Knowledge-Creating Organization introduced in Section 2, showing how this helps characterize new conversations in terms of their creation and articulation of new knowledge. This provides a way of characterizing not only the current state of an organization, but also potential transition steps.
- In Section 6.3, we suggest some criteria for selecting participants of new conversations.
- In Sections 6.4 through 6.6 we shift from the "What" and "Who" of new conversations to "How" to foster them. Section 6.4 offers additional principles and techniques for facilitating inquiry-based interactions; it also offers some theoretical grounding in why the techniques work the way they do, and suggests ways that you can tailor and tune them for your organization. Section 6.5 offers guidelines for scripting dramatic scenarios like the one in Section 4. Section 6.6 suggests some alternative techniques.

6.1 Key Concepts

Reuse Depends on Inquiry

Inquiry-based interactions are integral to systematic learning-oriented reuse. An organization's ability to sustain these patterns requires that all of the people involved in the network practice inquiry and learning-based interaction skills.

As an example, suppose that an application engineer searches a reuse library for a component and does not find the desired functionality. She could simply conclude from this failed search that the components are incomplete or of poor quality and abandon the library. Alternatively, she could decide to initiate an exchange with the component base manager, in which other possibilities are be explored. For example, she may discover that the scope of the component base was unclearly documented or misinterpreted, or that she used the search mechanisms improperly. She and the component base manager may also jointly discover that, although the functionality she requested was not present, it would enrich the component base if implemented. The quality of the component base and the degree to which it maintains coherence over time will directly reflect the quality of inquiry-based interaction skills among all stakeholders (component developers, brokers, and utilizers).

From Assessment to Adoption

In conventional approaches to technology transition, assessment is usually followed by reuse advocates selling reuse to management to persuade them to endorse a reuse program or project. The initial assessment and selling/buying interactions are closely coupled. In such an approach,

reuse adoption is often expected to be achieved through the advocacy of a lone "reuse champion" making a compelling sale to a senior manager, who is perceived (unrealistically) to have the power to enforce reuse by fiat.

The LIBRA assessment approach changes both these interactions. Reuse adoption is furthered less by a single advocate's persuasiveness than by creating an effective network of interactions among various sponsors, advocates, and practitioners. Reuse activities by their nature must involve individual initiative from an entire network of individuals at many levels of the organization.

Reuse "happens" in the course of knowledge-creating and knowledge-sharing interactions about software products and processes. The assessment process involves creating a forum for these interactions, measuring their effectiveness, and increasing their value. Self-assessment leads first to identification, then to stimulation or engagement of a network of interactions that may cross organizational boundaries, hierarchical levels, or departmental, functional, and project lines.

The network defines a minimal "unit of change" for an adoption effort. Attempts to introduce systematic reuse on a unit smaller than this network may not succeed because certain key roles or processes will be missing. Conversely, attempts to introduce reuse across a large division of the organization, independent of a distinct network of interactions, may prove non-optimal: change may be attempted in areas that are not essential for a first success, while essential aspects of the network that fall outside project or divisional boundaries will not be addressed. The result may be a failed or unsustainable initiative.

Choosing the Intervention

Even though change within a network of interactions may be tractable, no individual reuse proponent (whether a technology advisor in a consulting role or a manager with authority to set policy) can transform the network in one stroke. A more effective strategy is to identify a specific new interaction that is the key leverage point in creating movement towards the desired new pattern.

This new interaction can be thought of as a ***reuse intervention***, a shift in communication patterns that leads to a reuse-based network of interactions that persists over time. Finding the right leverage point for a given situation and desired goal-state is a substantial challenge for the reuse proponent. Indeed, it can be viewed as the core challenge in reuse adoption.

Interventions can vary widely in terms of the magnitude of the organizational change required; for example:

- Initiating interactions between people currently performing reuse (e.g., application developers with common needs begin to talk);
- Getting an organizational unit or an individual to shift their responsibility to include a new role (e.g., a developer takes on the role of asset broker);
- Creating a new organizational division or a new position linked with a needed role.

New Conversations

It can be helpful to think of these interventions as "new conversations." Of course, interactions can take many forms: electronic mail messages, management directives, requirements documents, and even software implementations. Direct, face-to-face conversations, however, are good contexts in which to observe and practice inquiry, because the elements required for dialogue are

most directly under the control of the participant. You can practice new interaction skills in direct conversation that, once internalized, can then be reflected more broadly in subsequent work interactions [Flor78].

6.2 Knowledge-Creating Interactions

The Knowledge Evolution Grid shown in Exhibit 20 (first presented in Section 2) is a useful framework for understanding how knowledge can be captured and made accessible in an organization. The grid can also be used as a tool for assessment and identification of possible new interactions. Following are some suggestions for its use.

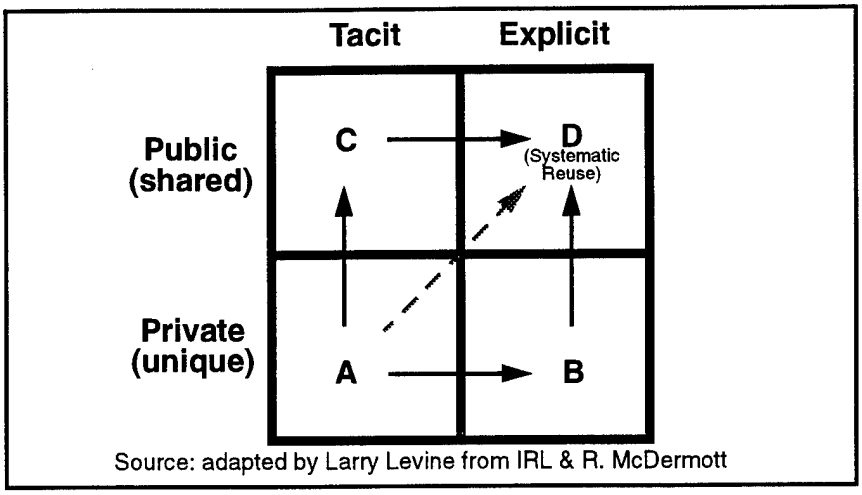


Exhibit 20. Knowledge Evolution Grid

6.2.1 Assessing Technology Competencies

The Knowledge Evolution Grid can be used to assess the current state of domain knowledge or competency within an organization. For example, suppose a company has developed a track record for successful implementation of hospital information management systems. With respect to a specific area of knowledge within this line of business (e.g., the design of the patient information database) the question could be asked: "Who knows how to do this?" or, phrased another way, "How is it that the organization knows how to do this?" Each quadrant of the grid corresponds to a different reply to this question:

- Quadrant A (Private/Tacit): The knowledge resides in the heads of a few key expert or veteran developers.
- Quadrant B (Private/Explicit): The knowledge has been codified in documents or software tools, but these artifacts or tools themselves still live in individual's desk drawers or personal directories. If anyone else wants to use them, they still need to know who the experts are and then go speak with them.
- Quadrant C (Public/Tacit): The knowledge of "how to do it" lives in the social network — in the culture, as it were — of a development group. (Such groups are typically small because this kind of knowledge creation, being un-systematic, does not scale up well to large groups.) When a job of this sort needs to get done, it is accomplished through a complex, largely infor-

mal series of communications, hand-offs, conversations, and messages among the members of the team. No one of them has the whole picture, and the whole picture is not written down anywhere.

This is the kind of knowledge that proves most ephemeral, when, for example, a software system is acquired and one or two experts follow along, but the sustaining social network that really kept it running is destroyed.

- Quadrant D (Public/Explicit): The knowledge has been codified in some kind of enterprise infrastructure. This is the state of affairs desired by most reuse advocates: for example, a company-wide library of reusable software is a classic way of making the knowledge both public and explicit.

  One reason such corporate-wide repository efforts have had limited success in the past may lie in a failure of reuse technologists to see the links between this quadrant and the others in the grid. For a large company with many lines of business and divisions, a corporate-wide repository may be the wrong level at which to direct efforts. The kind of information that makes it into such a repository may be information perceived as domain-independent (e.g., general utility libraries have been set up in most large software-intensive companies).[1] The information most critical to capture, however, may be much more local to different groups: the private-explicit knowledge codified in individuals' tools, or the public-tacit "ways of doing things" that grow in close-knit informal group interactions.

6.2.2 Assessing Interactions

The use of the Knowledge Evolution Grid involves diagnosing how, and to what extent, knowledge has been codified in a given knowledge area. The same grid can provide a framework for assessing the quality of interactions or initiatives within the organization.

One of the key insights that the grid diagram itself helps communicate is that it is difficult to shift knowledge in one step from Private/Tacit (Quadrant A) directly to Public/Explicit (Quadrant D). Such a step really involves two important bridging activities, and each has its own set of motivators and points of resistance.

Excluding this direct, diagonal path, we can consider four distinct shifts in the state of knowledge within the grid that are knowledge-producing or knowledge-creating:

- A -> B (from Private/Tacit to Private/Explicit): This shift takes place when individuals codify their own knowledge in more tangible form: writing it down, implementing it as a software tool, maintaining an informal library of code that they can reuse on multiple projects. Repeated shifts of this type are the basis for how experts or "gurus" develop their skills in particular areas. This kind of knowledge typically leaves with an engineer when he or she moves out of an organization.

- A -> C (from Private/Tacit to Public/Tacit): This shift takes place when team interactions are fostered within a group. The knowledge may be transferred in hallway conversations or scribbled on napkins in the cafeteria, but it can still be a powerful source of a group's productivity. Managers who understand the dynamics of this shift can do a lot to foster it. On the

---

[1] An empirical survey of how different companies have developed and maintained these central utility libraries would be a valuable contribution to the reuse field. How have they worked? How were they organized? What problems did they encounter? If such a library exists in your organization, you will want to know how to characterize a systematic reuse effort such as a domain engineering project in relation to this already existing facility.

other hand, these dynamics can be sabotaged by beliefs like "if people are standing around talking they can't be getting work done."

- B -> D (from Private/Explicit to Public/Explicit): This shift takes place when incentives are created for individuals to migrate their private reusable solutions and tools into some common infrastructure. For example, when a corporate reuse library is set up it might be assumed that contributed assets must be created from scratch. But in many cases, there will be prototype assets scattered throughout the organization that have never been moved from private to public status. Understanding this dynamic can help those who are initiating reuse efforts to tune incentives and qualification criteria appropriately.

  For example, one common pattern of failure (or undramatic success) in reuse libraries is that people will readily place their own components in the library, and then will reuse them out of the library rather than out of their own private directory. But they will not use other people's components, nor will they make efforts to see that their own components are used by other people. These patterns can be measured and addressed more readily if it is understood that the library is being used as an intervention to move knowledge from Private/Explicit to Public/Explicit form.

- C -> D (from Public/Tacit to Public/Explicit): This shift occurs when knowledge that is held informally in the network of interactions is codified. This dynamic may typically combine the organizational learning disciplines Senge refers to as *team learning* and *mental models*. The process of converting shared but unarticulated knowledge to something repeatable, transferable beyond the group, or amenable to automated support, requires that people work together in teams to learn and codify their learning, and not just to "get the job done." (Or rather, to get a different kind of job, the "learning job," done.)

  Domain modeling, treated as an activity that involves full participation of the domain practitioners (which ideally involves both the developers and the users of software systems in the domain), is a process designed to facilitate this shift. If carried out in the context of the Knowledge Evolution Grid, domain modeling has the advantage that the eventual domain assets produced are created and owned by the people who will use them to build applications. Gathering domain data through activities that bring groups of developers together (rather than analyzing artifact analysis or conducting one-on-one interviews with experts) has an additional benefit: it generates new knowledge by bringing individual experience bases and perspectives together in conversation. Such activities strengthen the sense of technical community within the group. Only this community can sustain the asset base, treating it as their own "transferred technology."

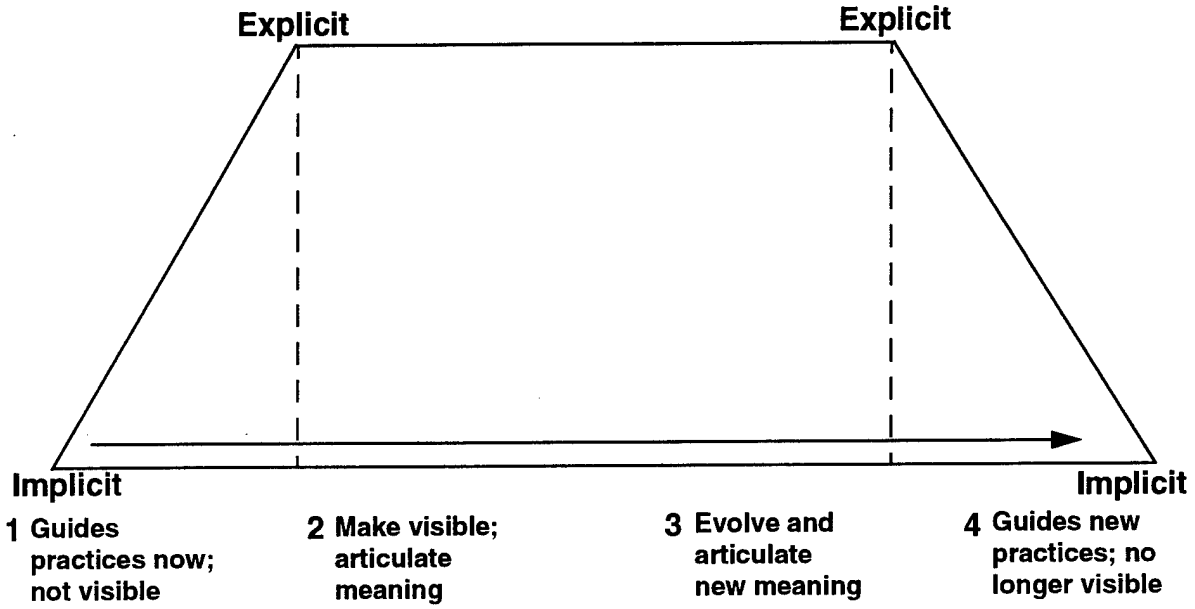
These four separate shifts form two distinct paths towards optimal knowledge sharing and knowledge creation in organizations. The greatest flexibility exists when the current state of the knowledge is both private and tacit. In such a situation, the model suggests that an incremental strategy makes good sense: either encouraging public sharing of tacit knowledge in an informal setting, deferring the pressure to codify the knowledge explicitly until later in the process; or, conversely, encouraging codification (letting a thousand flowers bloom) before trying to enforce standards and common sharing of the knowledge assets produced.

6.2.3 The Knowledge Creation Life Cycle

The Knowledge Evolution Grid is a useful framework for understanding how knowledge migrates from private and tacit to public and explicit. However, the grid does not in itself address the strategic competitive value of the knowledge being assessed. Thus, used in isolation, it can be misleading; every organization must determine which areas of knowledge are critical to its core business and which are ancillary or peripheral. Since knowledge creation and systematic learning

require commitment of resources, it is not sufficient to simply say that the ideal knowledge-creating organization is creating knowledge in all areas of work practice, with equal priority, all the time.

A related insight is that organizations do not work efficiently by leaving knowledge at its most public and explicit all the time. For any given area of knowledge there is a ***life cycle***, illustrated by the "trapezoid" diagram in Exhibit 21 [Nona95]. The trapezoid illustrates the shift over time between **implicit** (or "tacit") and **explicit** knowledge. (The same basic pattern could apply to the movement between private and public knowledge.) The picture emphasizes the idea that knowledge does not remain indefinitely in a fully explicit state. Knowledge is made explicit for particular purposes; then, over time, there is a natural tendency for the knowledge to become "compiled down" back into implicit form. This re-submergence is an indication that the knowledge has become deeply internalized and shared.



Source: Adapted by Larry Levine from Nonaka

Exhibit 21. The Knowledge Creation Life Cycle

The fact that work groups get anything done is testimony to the fact that most of the knowledge we employ in any technically intensive task remains implicit. This includes the shared technical culture, the contextual assumptions that mean we do not have to define terms from scratch with each interaction, and the accepted way of doing things that leads to smoothly running, competent organizations. Some reasons to make knowledge explicit include the following:

- To codify it and make it more systematically reusable;
- To transfer it to different groups or individuals;
- To reduce dependency on particular experts for maintaining the knowledge;
- To validate the knowledge and verify that it is still relevant in a changing business and technical environment;
- To be able to re-examine and evolve the knowledge.

The last point is key: knowledge needs to become public and explicit before it can be strategically evaluated and changed. It can then gradually be re-integrated into the organization's tacit knowledge base.

Using the Knowledge Evolution Grid and the Knowledge Creation Life Cycle in combination, it should be possible to ask the following questions for any domain or area of competency:

- How strategic is this knowledge to our organization? Is it a core competency or a means to an end, a way of getting something else done? Does it contribute to our competitive advantage?
- Where does this knowledge reside in the Grid? How public and explicit is it in the current environment?
- What is the current trend within the organization? Are there initiatives in place that are moving the knowledge towards more codified form? Conversely, is a previous codification effort beginning to "decay" towards private and implicit forms of the knowledge again?
- How desirable is the current state, and any current movement? Is the movement in accord with the current life cycle stage? Is it an appropriate time to initiate a new knowledge creation life cycle for this domain?

By addressing these questions, it is possible to focus efforts and resources on knowledge creation initiatives that will produce the maximum strategic benefit for the organization. This could also lead to effective diagnosis of efforts that are underway but are facing significant business obstacles. For example, if a large-scale company standardization effort were underway in a technical area that is undergoing rapid evolution and instability, this assessment might help make clear that the effort is premature (or too late to be effective).

6.2.4 A Knowledge Creation Belief Map

The Knowledge Evolution Grid and Knowledge Creation Life Cycle, taken together, help reveal a belief map that can influence, and sometimes compromise, reuse adoption strategies. This belief map concerns the value placed on organizational change vs. organizational stability:

- Stability is Compulsive: Change is generally good. People in organizations who resist change are compulsively sticking to "the way we've always done things" even though a new way might be better. Stability works against the adoption of new ideas and new technology. Reuse is a new idea, supported by new technology. Adoption of systematic reuse therefore requires that we break down people's compulsive sticking to old practices.
- Change is Impulsive: Change for its own sake is often not good. People in organizations who advocate change as if it were its own motivation are often not concerned about the health of the organization at all. They are, rather, more concerned with their own career and power within the organization, since being an agent of change generally increases one's influence during the change process. Even if the motives are not so devious, the impetus to change is often a reaction to the latest management fad.

The two paragraphs above articulate two belief "clusters"—sets of related, reinforcing beliefs, often with chains of inference among key beliefs. These two belief clusters approximate the stances of "reuse technology advocates" and those they perceive as "bastions of resistance" within the organization. As is common with beliefs active in a community, they form a clear polarity. Interactions based on advocacy and persuasion might only serve to reinforce each position more strongly and diminish chances for dialogue.

Building on our two knowledge creation tools, we suggest a third, intermediary belief that bridges these opposing stances:

> It is important for an organization to make strategic decisions about when to initiate knowledge creation cycles in particular domains. Only when such a cycle has resulted in public and explicit representations of the knowledge can the organization decide whether to shift or evolve the knowledge. We should neither cling to old techniques compulsively nor abandon them impulsively; similarly, we should neither institute nor avoid change for change's sake. Systematic reuse practices can help us raise embedded knowledge to awareness, to validate it and make it strategic, shareable, and reusable.

A person acting out of this belief will be able to converse effectively with people advocating either of the polarized beliefs. This makes it a useful belief for someone attempting to identify opportunities for reuse within their organization.

6.3 Selecting Participants for New Conversations

The CFRP process model, the Organization Domain Modeling (ODM) domain engineering method, and the knowledge creation and evolution roles presented in Section 5 offer approaches for identifying participants for new conversations — changing the "Who" in interactions. These ways of identifying multiple stakeholders may be difficult to apply in initial, small-scale assessments. The following list offers examples of criteria to aid in selecting potential participants in new conversations:

- Lateral relations: identify people who perform analogous roles on different projects. For example, bring people who have done regression testing on different projects together to find out whether there are common needs, tools that have been developed, sharable artifacts such as test data, etc. This is related to the *quality circle* approach from Total Quality Management (TQM), but the principle can be applied informally on a smaller scale.

  If there are established forums for this kind of interchange, there may be a positive climate for reuse in the organization. There may also be structures in place that provide this communication function without face-to-face meetings (e.g., Lotus Notes discussion groups, company e-mail, informal networks of information exchange among employees).

- Hand-off relations: Identify people who are "along the pipeline" in terms of workflow. The farther apart the people are, the less likely that they will have had opportunities for direct contact as part of ongoing project work. This is particularly true when such pipelines cross organizational boundaries (vendors, suppliers, customers).

  This principle is related to the concept of just-in-time delivery (e.g., close communication between component suppliers and integrators), business process reengineering (reengineering "whole processes" grounded in clear customer value), systems thinking (getting the whole system perspective) or concurrent engineering (transforming sequential to parallel engineering processes).

- Diverse views: Try to get as diverse a set of stakeholders together as possible around a given topic. Here the goal is learning from multiple perspectives. This approach is an important element of *search conferences* [Weis95], which tend to be applied in large-scale settings. For example, a search conference for an organization might draw together employees, customers, competitors, members of the surrounding local community, representatives of regulatory

agencies, etc. Such a diverse cast of characters would not be appropriate for a typical decision-making meeting, but could be an opportunity for significant learning.

- Structural relations: Most meetings in an organization are either interactions between peers, between people one hierarchical level apart (e.g., managers with their direct reports), or events like annual meetings where the CEO lectures to everyone. Direct communication between people removed by more than one level can be threatening to the authority structure under normal circumstances (e.g., occasions for micro-management on the part of high-level managers, or side-stepping the chain of command on the part of lower-level employees). However, a meeting that involves three or more structural levels, or both managerial and technical personnel, or both technical and marketing staff, could create significant opportunities for creation of new knowledge.

- In selecting participants for an interaction, consider the potential impact of who is *not* there as well as who is. For example, a meeting with a division manager and front-line supervisors would have a very different dynamic if the person at the intervening level of reporting (e.g., the project manager) is or is not present at the meeting. Presence or absence of organization "outsiders" will also have a dramatic effect on the interactions.

These are only a few examples of guidelines for selecting participants. Besides these general criteria, every organization will have unique association patterns, people who often talk with each other and people who rarely do; e.g., veterans who joined the company during one era may not interact with newcomers, and so on. Initiating assessment conversations that cross such boundaries can be a significant intervention in the communication patterns of the organization.

6.4 Additional Tools to Facilitate Inquiry

The preceding sub-sections provide guidelines for identifying possible new conversations, choosing participants for those conversations, and combining assessment with the conversations.

In order for these conversations to serve as transition steps towards learning-oriented reuse, it is vital to conduct the conversations in a way that fosters inquiry. Inquiry skills are essential to the LIBRA approach to reuse assessment. By using these techniques during assessment, you are simultaneously *learning* skills that will be needed for reuse, *exemplifying* those skills through your own behavior, and *transferring* them to others by facilitating new interactions.

The Ladder of Inquiry presented in Section 3 is a tool for assessing interactions in terms of inquiry vs. advocacy. The following sub-sections present some techniques for fostering inquiry:

- Listening and building rapport;
- Engagement vs. cooling;
- "Indirectness" as a cooling strategy.

6.4.1 Listening and Rapport

Several core skills can be applied to increase the level of inquiry in a given conversation:

- ***Tolerance*** — a willingness to let other people hold and articulate positions that are not aligned with ours.
- ***Openness*** — a willingness to disclose our own beliefs and positions.

- ***Reflectiveness*** — a willingness to explore our own motivations, assumptions, and tacit beliefs. When we engage in inquiry we are willing to learn something about our own attitudes as an outcome of the conversation.
- ***Dialogue*** — a willingness to explore the reasoning behind another's stated positions, even (or especially) when those positions are not aligned with our own. Dialogue involves asking questions in a non-challenging way that invites authentic responses, and imagining and reasoning about another person's experiences, beliefs, and positions. (The latter is as important as the direct asking of questions because it is not always possible to ask directly.)

These skills are interrelated, as shown in Exhibit 22. Different skills are needed when dealing with one's own positions and beliefs (*Self*) versus those of others (*Other*); and when dealing with tacit or hidden positions and beliefs (*Disclosure*), versus those that are available to examine (*Discovery*).

Good ***listening*** is an essential prerequisite for all the above skills. *Passive listening* is most closely related to tolerance, simply giving people the chance to speak and feel heard. *Active listening* involves openly exploring and validating what others are saying by paraphrasing their words and getting their confirmation or correction.

Listening skills are essential in dealing with what *is* getting said, but additional skills are needed to build ***rapport***. These skills, practiced by skilled interviewers, reporters, and facilitators, involve creating a general climate of trust and using observation and insight to discover what is *not* being said.

This informal picture of inquiry-related skills can be used to assess one's own skill level, analyze the quality of an interaction, or identify where breakdowns are occurring.

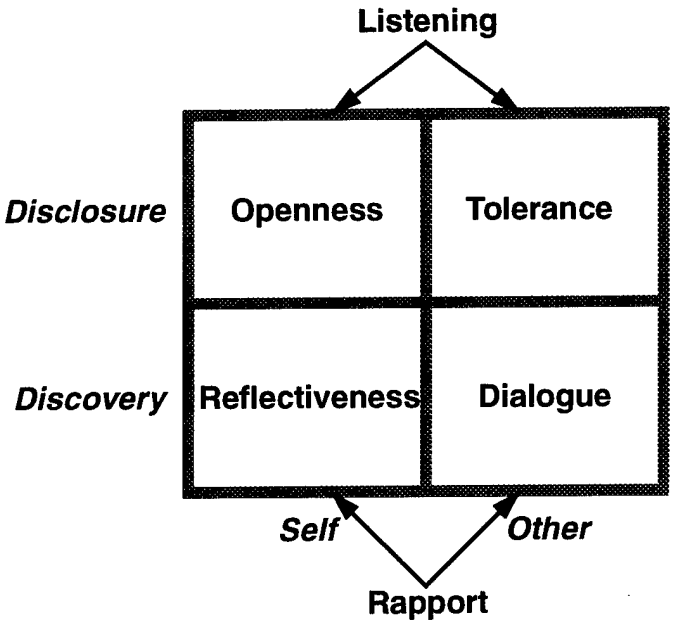


Exhibit 22. Relationships among core inquiry skills

6.4.2 Engagement vs. Cooling

Even when the above skills are practiced well, inquiry-based interaction can be difficult to achieve. One explanation for this is provided by David Bohm [Bohm90]. Bohm distinguishes two aspects of interactions that can vary independently. The one most familiar to us on an informal basis might be called the "quality of discourse" or "emotional temperature": how "hot" or "cool" the conversation is. (Our expression "heated discussion" conveys the sense pretty directly.)

What makes a given conversation tend towards the hot or cool? Bohm draws attention to the degree of interest or engagement that participants have in the topic of conversation. The interaction of these two factors is depicted in Exhibit 23.

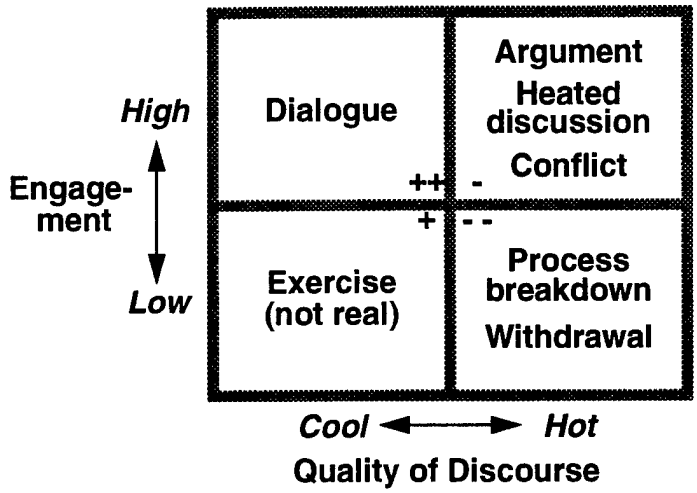


Exhibit 23. Relationship between Engagement and Quality of Discourse

As the diagram illustrates, it is no great challenge to stay cool about a topic in which one is not particularly interested. This quadrant (Low Engagement/Cool Discourse) is best exemplified by a group of bored students dozing in a classroom. The problem comes as our engagement increases. If inquiry skills are not being applied, the tendency is for the heat of the discourse to increase with the degree of engagement (moving along the diagonal to the High Engagement/Heated Discourse quadrant). If things get bad enough and the process breaks down, the heat remains but the engagement wanes (Low Engagement/Heated Discourse). Process breakdowns will ensue over smaller and smaller issues; or one or more parties may completely withdraw from the interaction.

The challenge in inquiry, then, is how to get to the upper left quadrant (High Engagement/Cool Discourse). How do we reason together about things that matter a great deal to us? There are two paths that lead most directly to this state. Moving directly from argument to dialogue means overcoming a greater "differential," and is the kind of situation where outside facilitators or conflict resolution consultants are called in. For self-assessment situations, where people are trying to manage their own meetings and develop their skills incrementally, a safer path is a gradual shift from Low Engagement to High Engagement, keeping the discourse Cool throughout the process.

This diagram can be used directly as another monitoring tool for assessing the current state of an interaction (e.g., a meeting, a negotiation) and for trying to direct it. The diagram also provides some rationale for applying scenario-based techniques in assessment. In constructing an overall approach to assessment, various strategies for "cooling" interactions can be considered. One primary way of moving adaptably along the "engagement axis" involves intentional use of indirectness techniques, described in the next section.

6.4.3 Indirectness Techniques

Why is the issue of hot vs. cool discourse relevant to reuse assessment and reuse adoption? Many barriers to improved reuse practice within organizations touch on sensitive behaviors and tacit beliefs. These are not easy to address. They may strike at deep layers of cynicism and frustration. Direct confrontation on these issues may not only be ineffective, but may entail considerable risk.

Thus there are advantages to techniques that make opportunities and barriers to reuse visible in indirect ways. This is typical of highly professionalized knowledge-intensive fields. The work of Chris Argyris helps us to understand a classic double-bind:

- Professional knowledge workers have strong defenses against learning from their own practice and experience;
- These defenses cannot be easily challenged, because they are "self-sealing:" confrontation about the defensiveness activates the very defensive patterns that are being questioned.

How do we get out of this double-bind? One technique Argyris suggests is described in his article, [Argy91]. A CEO writes a "script" for a staff meeting he intends to hold; he includes dialogue for the obstacles he expects to face and his own internal thoughts during the interchange. Then, instead of holding the meeting described in the script, he holds a meeting where the participants reflect on the script itself. Somehow, this level of indirectness allows reflection to occur where it had been prevented before.

In Argyris's case study, it is open to interpretation whether indirectness was a key element allowing forward movement. There is a well-recognized human tendency to accept information about highly sensitive topics through some appropriate veil, or mask, or projection. (Most doctors are probably familiar with the opening line, "I have this friend...") This technique could also be called *distancing*.

Guidelines

Based on our insights about engagement vs. cooling and directness vs. distancing, we can return to the ladder metaphor embodied in the Ladder of Inquiry and imagine a "Ladder of Directness," where pure confrontation is at the top and discussion of purely hypothetical situations is at the bottom. This ladder reflects the following heuristics:

- Narration is cooler than dramatization, and theory or abstract discussions are cooler than narration of a specific scenario.
- People will resist abstract presentations of concepts (e.g, the AMEGO or "And My Eyes Glazed Over" phenomenon). They will almost always say that they prefer concrete examples. But if these examples touch on sensitive issues, increased engagement and heated discussion may interfere with reflection and listening.
- Working with pre-defined materials is "cooler" than having people generate their own material.
- Working with hypothetical material is generally "cooler" than examples or real case studies.
- Working with examples from other organizational settings is generally cooler than examples drawn from the organization conducting the inquiry.
- Humor can be a technique that cools down hot topics and makes them more discussable.

- In general, the more "distanced" the elicitation technique, the "cooler" and more easily facilitated will be the interactions, but the more expertise will be required to interpret the data derived from the exercise.
- Conversely, the more direct the elicitation technique, the "hotter" and potentially risky will be the interactions, and the greater will be the required facilitation expertise. However, interpretation of the data may be easier, with less translation required.

Despite the symmetrical heuristics outlined above, the ladder metaphor is only a convenience and may be misleading in several important respects:

- It is not always the case that interactions will be automatically "cooled" through strategies of indirectness. For example, dramatization techniques, even of hypothetical case studies, can lead to emotionally charged interactions. They can evoke responses at a symbolic and nonverbal level that may be difficult to anticipate or control. There is always a potential risk in facilitating such interactions and such a task should not be attempted lightly. (See the discussion below on running your own dramatic scenario scripting session.)
- Unskilled facilitation of a "hot" elicitation exercise could result in "overheating" and poorly resolved conflict; or it could result in apparently sound results that actually provide inaccurate or masked data. Unskilled interpretation of a "cool" elicitation could result in significant distortions of the data. Most important, such situations can evoke strong personal reactions difficult to predict in advance. We do not recommend that anyone attempt these approaches without respect for the risks involved and access to skilled facilitation expertise.

6.5 Scenario Scripting and Interpretation Techniques

In the earlier sections of this document, we provided examples of the *results* of inquiry-based techniques such as dramatic scenarios, system diagrams, and belief maps. In addition, the scenario creation (or "scripting") and interpretation processes themselves can play an important role in learning-oriented self-assessment.

Scripting the Scenario

In the scripting activity, people individually or collaboratively develop the core materials for a scenario. These materials could include any of the following:

Character profiles. For each character, the profile should include the following:

- Name, age, job title or position
- Salient beliefs, including:
  - What is expected of me in my role(s)?
  - What resources do I need to meet those expectations?
  - What do I expect of other players?
  - What is "goodness" in my job (pleases/satisfies me)?
  - What is "badness" in my job?
- Resources available to me (technical or other) to meet my objectives?

- My situation: anything else about the job context relevant to how this character will perform; in particular, any potential sources of "dramatic conflict";
- The character **spine**: a concise statement of what this character is "all about" in the drama (not necessarily the same as the official job role!)

Scene Descriptions. For each scene, provide the following information:

- What is the "objective" (theater and film directors call this the "throughline") of the scene?
- What characters play a role in this scene?
- What is each character's main objective (overt or tacit?)
- What conflict or challenge motivates the scene?
- What are the major interactions?
- How is the conflict or challenge resolved?

The scenes are generally organized into acts, and the scenario (or drama) as a whole also has a throughline as well as a detailed plot.

There are a many ways to organize the generation of this material. Possible strategies include working alone or individually, preparing material in advance of a meeting or working through the material at the meeting, and doing round-robin critique and commentary on each other's work.

Interpreting the Scenario

The scenario material above can be analyzed and interpreted in a variety of ways to yield deeper information about the characters, organization, and events. Useful focal points for such interpretation include:

Character Belief Maps. List the key "belief issues" faced by characters in the scenario. For example:

- *Explicit stateable beliefs:* What are the beliefs that form the explicit, stateable shared basis for action among the characters? In our example, one such belief might be: This is a for-profit organization whose primary work is in the government sector. (Business models persist in organizations because all participants share a set of beliefs about the kind of work they can pursue and its desired outcomes.)
- *Explicit but unstateable beliefs:* What are the things that everyone knows but no one can actually say in meetings? This could include harsh business realities such as, "Our competitors are eating us for lunch." In one organization this might be openly discussed and used as a basis for planning; in another it may be a point that is hotly denied at managerial or technical levels in the organization.
- *Implicit/tacit shared beliefs:* What beliefs or mental models are so ingrained that people don't recognize them as shared beliefs or bases for action? These are more likely to be beliefs held by the organization or broader culture. Cross-cultural contact may help make these beliefs and assumptions explicit.
- *Key conflicts/polarities in belief:* Communities are characterized as much by conflicts or polarities as by consensus. Some of these can be ongoing debates, others will flare into criti-

cal tension around key decisions that need to be made (e.g., Do we diversify our line of business? What business are we in?)

Organization Spine. Treat the organization itself like a "character" by describing the overall business environment.

Organization Beliefs. As with the beliefs held by individual characters, identify the commonly held beliefs and the key polarities faced within the organization in terms of those which are: 1) public or open; 2) acknowledged but unspoken; and 3) invisible to the characters. Clarify the characters' stances with respect to key belief polarities in the organization.

Key Events. Each key event in the scenario can be used as the focus for a facilitated inquiry session. For each key event, questions could include the following:

- What are the key issues faced by the characters?
- What key themes are emerging?
- Was an opportunity lost here?

Relationships Between Key Events. Scan the scenario for key events or interactions that reveal unspoken polarities in beliefs: moments where the invisible or undiscussable become accessible.[2] Look for relationships between the key events. Questions that can trigger dialogue include:

- *Backward threads:* What premonitions or previous clues in the scenario foreshadowed the issues active in this event's dramatic situation?
- *Forward threads:* What are the repercussions or consequences of this event later in the scenario?
- *Mirroring:* What other events seem to reflect the same issues in different settings? Could these be the effect of relationships between the characters? (For example, managers receiving pressure from above tend to reflect that pressure down onto subordinates; companies may tend to replicate their relations with suppliers in their supplier role to their customers.)
- *Echoing:* If there is no clear causal connection, we call the relation *echoing*. Echoes might signify organizational cultural patterns (part of the belief map for the organization).

Running Your Own Scenario

The scenario interpretation technique offered in this document is designed to be adapted to various rungs along the "Ladder of Directness." The ladder might include the following "rungs," moving from greatest to least distance:

- Use the pre-defined case study as a basis for discussion.
- Modify the case study as required.
- Create your own case study which does not necessarily reflect the current organizational situation.

---

2. Movement in this direction seems to be inherently dramatic. Movement from public to unspoken, from unspoken to invisible, tends to happen over longer periods of time, as a gradual "forgetting" process. This is related to the later phases of the Knowledge Creation Life Cycle.

- Create a "generic" or "anonymous" case study based on your own organizational environment (a fable).
- Create a script involving your direct situation and discuss (but do not enact) the script.
- Create a comical farce about reuse in your organization and perform it at an annual meeting.
- Create a deadly serious tragedy about reuse in your organization; perform it; update your resume.

6.6 Other Inquiry-Based Techniques

Some alternative tools and techniques that can be used to facilitate new conversations are:

- Biography (first person narrative, from the viewpoint of an individual or, in some cases, from the viewpoint of an organization as a whole);
- Project history (a narrative at the organization level, with material collected and presented at least in part through interaction with an outsider's perspective);
- Journal (e.g., design journal, process history, maintained iteratively throughout the performance of a given project);
- Dramatization (enactment in face-to-face interactions of the dynamics of a given situation);
- Workplace ethnography (a combination of direct observation of practice, analysis of artifacts, and qualitative interviewing by skilled outsiders);
- Search conferences (facilitated meetings that involve as diverse a set of stakeholders as possible to create common ground and envision future possibilities);
- Domain modeling as embodied in methods such as ODM (the cognitive processes and the social interactions involved in building comparative models);
- Formal process modeling.

The decision to apply a given technique should weigh a number of factors:

- The commitment of necessary resources;
- The degree of skill required to facilitate them;
- The degree to which outsiders' perspectives are needed to derive value from the approach (e.g., an external consultant working with a group);
- The degree of risk a technique may entail regarding personal interactions.

References and Recommended Reading

The following list includes publications that are referenced in the document as well as selected publications recommended for further reading.

[Argy91] Chris Argyris. "Teaching Smart People How to Learn." *Harvard Business Review*, May-June 1991.

[Barn91] Bruce Barnes and Terry Bollinger. "Making Reuse Cost-Effective." *IEEE Software*, vol. 8, no. 1, January 1991, pp. 13-24.

[Berl90] L. Berlin. "When Objects Collide." *Proceedings of the Conference on Object-Oriented Programming: Systems, Languages, and Applications/European Conference on Object-Oriented Programming (OOPSLA/ECCOP '90)*, Ottawa Canada, October 1990, pp. 181-193, ACM Press/Addison-Wesley Publishing Company.

[Bloc81] P. Block. *Flawless Consulting: A Guide to Getting Your Expertise Used.* Pfeiffer & Co., San Diego CA, 1981.

[Bohm90] D. Bohm. On Dialogue. David Bohm Seminars, Ojai CA, 1990.

[CARD92] Central Archive for Reusable Defense Software (CARDS). Acquisition Handbook. Unisys STARS Technical Report STARS-AC-04105/001/00, Reston VA, October 1992.

[CFRP93] Software Technology for Adaptable Reliable Systems (STARS). STARS Conceptual Framework for Reuse Processes (CFRP), Volume I: Definition, Version 3.0. Unisys STARS Technical Report STARS-VC-A018/001/00, Reston VA, October 1993.

[Clur72] Harold Clurman. *On Directing*. Collier Books, Macmillan Publishing Company, 1972.

[Cox90] Brad Cox. Planning the Software Industrial Revolution." *IEEE Software*, vol. 7, no. 6, November 1990, pp. 25--33.

[Crui91] R. Cruickshank and J. Gaffney. The Economics of Software Reuse. Software Productivity Consortium Technical Report SPC-92119-CMC, September 1991.

[DoD92] DoD Software Reuse Initiative. DoD Software Reuse Vision and Strategy. Center for Software Reuse Operations, Technical Report 1222-04-210/40, Alexandria VA, July 1992.

[Flor78] F. Flores. Management and Communication in the Office of the Future. Doctoral Dissertation, U. C. Berkeley, Berkeley CA, 1978.

[Gaff89] John Gaffney and Thomas Durek. "Software Reuse -- Key to Enhanced Productivity: Some Quantitative Models." Information and Software Technology, Vol. 31, 1989. Also Software Productivity Consortium Technical Report SPC-TR-88-015, April 1988.

[Gris95] Martin Griss. "Software Reuse: Objects and Frameworks are not Enough." *Object Magazine*, February 1995.

[John95] Ralph Johnson. "Why Don't Reuse People Talk about Reusable Software?" *Proceedings of the 7th Workshop on Institutionalizing Software Reuse*, St. Charles IL, August 1995.

[Laur93] Brenda Laurel. *Computers as Theatre*. Addison Wesley, 1993.

[Lato90] Larry Latour, Tom Wheeler, and Bill Frakes. "Descriptive and Predictive Aspects of the 3Cs Model: SETA1 Working Group Summary." *The First Symposium on Environments and Tools for Ada*, Redondo Beach CA, May 1990.

[Levi94] Larry Levine. "Listening with Spirit and the Art of Team Dialogue." *Journal of Organization Change Management (JOCM)*. Vol. 7, No. 1, 1994, pp. 61-73.

[Lind93] C. Linde. "Reflections on Workplace Learning." Working Paper, Institute for Research on Learning, Palo Alto CA, 1993.

[Moor91] Geoffrey Moore. *Crossing the Chasm: Marketing and Selling Technology Products to Mainstream Customers*. Harper Business, New York NY, 1991.

[Mcde95] R. McDermott. "Designing and Improving Knowledge Work." *Journal for Quality and Participation*, March 1995, pp. 72-77.

[Nona91] I. Nonaka. "The Knowledge-Creating Company." *Harvard Business Review*, November-December 1986.

[Nona95] I. Nonaka and H. Takeuchi. *The Knowledge-Creating Company, How Japanese Companies Create the Dynamics of Innovation*. Oxford University Press, New York NY, 1995.

[ODM95] Software Technology for Adaptable Reliable Systems (STARS). Organization Domain Modeling (ODM) Guidebook, Version 1.0. Unisys STARS Technical Report STARS-VC-A023/011/00, Reston VA, March 1995.

[RAG93] Software Productivity Consortium. Reuse Adoption Guidebook. Software Productivity Consortium Technical Report SPC-92051-CMC, Herndon VA, November 1993.

[Rief91] Donald Reifer. "The Economics of Software Reuse." *Proceedings of ISPA Thirteenth Annual Conference*, New Orleans LA, May 1991.

[RSM93] Software Technology for Adaptable Reliable Systems (STARS). Reuse Strategy Model: Planning Aid for Reuse-based Projects. Boeing STARS Deliverable D613-55159, Seattle WA, July 1993.

[Sche85] E. Schein. *Organizational Culture and Leadership*. Jossey-Bass Publishers, San Francisco CA, 1985.

[Sche93] E. Schein. "On Dialogue, Culture, and Organizational Learning." MIT Working Paper, Sloan School of Management, MIT, Cambridge MA, 1993.

[Scho83] Donald Schon. *The Reflective Practitioner: How Professionals Think in Action*. Basic Books, New York NY, 1983.

[Seng90] Peter Senge. *The Fifth Discipline*. Doubleday/Currency, New York NY, 1990.

[Seng94] Peter Senge, et al. *The Fifth Discipline Fieldbook: Strategies and Tools for Building a Learning Organization.* Doubleday/Currency, New York NY, 1994.

[Weis87] M. Weisbord. *Productive Workplaces.* Jossey-Bass Publishers, San Francisco CA, 1987.

[Weis92] M. Weisbord. *Discovering Common Ground.* Berrett-Koehler, San Francisco CA, 1995.

[Weis95] M. Weisbord and S. Janoff. *Future Search.* Berrett-Koehler, San Francisco CA, 1995.

[Wino87} T. Winograd and F. Flores. *Understanding Computers and Cognition: A New Foundation for Design.* Addison-Wesley, Reading MA, 1987.